Soliton automata: a computational model on the principle of graph matchings

Ph.D. Thesis

by

Miklós Krész

Advisor:

Dr. Miklós Bartha

Associate Professor

University of Szeged

2004

Contents

List of Figures

Preface

Molecular computing is an emerging field in current theoretical and application-oriented research ([1],[87], [101])as well. There are several research lines towards the common fascinating goal: to build a molecular computer. The present interest in biocomputing is due to many factors, but the most fundamental one arises probably from electronics. Limits are extended year after year, yet at some point the size, speed and power dissipation of switches based on silicon or other conventional materials will run into the deadlocks set by the basic laws of physics. One of the most promising alternatives of the traditional semiconductor technology is the so-called biolectronics or molecular electronics ([88]).

The idea of molecular memories goes back to the early science fiction of the 1930's and this way the concept of molecular structures to vast amounts of information is not new. The brain neatly proves that vast chemical information stores are possible. Feynman (cf. [44]) raised serious interest in molecular engineering with his pioneer paper, in which he proposes building small machines and then using those machines to build still smaller machines and so on, down to the molecular level. The idea of molecular electronic device was proposed by Aviram and Ratner ([5]), which was followed by an emerging interest in the new field in the 1980's ([26],[28]). Several different approaches were taken; for example, devices based on biological systems were proposed by several researchers such as Adleman([3]), and Conrad ([31]).

Other alternatives for molecular electronic devices were strongly based on the design of conventional digital circuits on the molecular level ([25]). The idea of this approach is: if we build up the electronic elements chemically from the molecular level, it would be possible to make circuits thousands of times smaller. These molecular circuits would use chemical molecules as electronic switches and be interconnected by some sort of ultra-fine conducting wires. One interesting possibility of these conductors was proposed by Carter ([27]) and is about using single strands of the electrically conductive plastic polyacetylene. Electrons are thought to travel along polyacetylene in little packets called *solitons*. Hence, molecular scale electronic devices constructed from molecular switches and polyacetylene chains are called *soliton circuits*.

Polyacetylene consists of a chain of carbon atoms held together by alternating double and single bonds (see Figure 1). Each carbon atom is also bonded to a hydrogen atom. Polyacetylene has two stable states, which differ in the position of the alternating double and single bounds with respect to the carbon atoms. A soliton is a moving wave which causes conversion between the two states of polyacetylene. It effectively picks up one arrangement of bonds and lays down the other. The effect of solitons on the state of a polyacetylene chain is shown by the arrows in Figure 1. The small curved arrows represent the movement of a pair of electrons. The combination of all these movements is the passage of a soliton, which is shown by the large wavy arrows. In a soliton circuit all the chemical components are interconnected by single strands of polyacetylene. Solitons through these components will cause changes in their state, in much the same manner, as a soliton changes the state of polyacetylene.

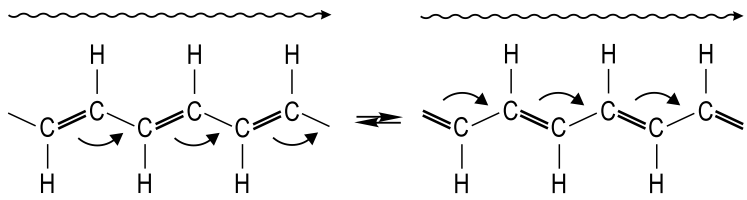

Figure 1: Solitons in polyacetylene

We note that the word soliton (”solitary wave”) is applied to many other types of waves traveling relatively large distance with little energy loss. During the last years, several additional computational models based on the soliton effect have been proposed. For a survey see [2].

This thesis deals with the mathematical model of soliton circuits called soliton automata. This model was introduced by J. Dassow and H. Jürgensen in 1990 ([34]) in order to capture the logical aspects of the ”valve” effect by which soliton switches and soliton circuits might operate. The underlying object of a soliton automaton is the so-called soliton graph representing the topological structure of the corresponding molecule-network. In this model atoms (or groups of atoms) are represented by vertices and chemical bonds correspond to edges. The vertices with degree 1 are designated as external vertices, while a vertex with degree greater than one is called internal. External vertices correspond to the marginal parts of the system, which parts serve as electron donors or acceptors for the remaining part of the molecule. Considering a computational structure, the external vertices serve as an interface to the outside world, through which the corresponding molecule-network as a computing device can communicate or be interconnected with other systems. The multiplicity of bonds (single or double) is fixed by a weight assignment to the edges of the underlying soliton graph. However, the internal vertices correspond to an atom (or group of atoms) with the property that among its neighbors there exists a unique one to which it is connected by a double bond. The states of the system are the weight assignments satisfying the above conditions, while state-transitions are realized by walks connecting external vertices and alternate on single and double bonds in such a way that the status of each edge during the walk is exchanged dynamically step by step.

Dassow’s and Jürgensen’s introductory work was followed by a series of papers (cf. [35], [36], and [37]) in which special cases of deterministic soliton automata were analyzed with respect to their transition monoids. Concerning another aspect of the computational power of deterministic soliton automata, in [58] a detailed analysis was given for homomorphically complete systems of these automata. However, no detailed theory has been developed for the description of the underlying topological structure of these automata, which explains the lack of more general results on soliton automata.

Parallel to the above theoretical work, the practical research funded by Circadian Technologies resulted a significant progress. As reported in a series of papers ([60], [61], [62], and [63]) appropriate chemical structures can be given for electronic devices which can interact with solitons. These results provided the first detailed look at how molecular electronic devices can be interconnected to form a variety of digital circuits. The need to develop an applied mathematical arsenal for studying soliton circuits has arisen quite early in the course of this research (cf. [59]), in order to obtain a detailed understanding of the behavior of these circuits. The most fundamental problems, like in circuit theory in general is the verification of these circuits motivated by the practical demand that a circuit can be mathematically verified for its possible use before attempting to build this. Nevertheless, apart from the early work by M. P. Groves, such a structural analysis has not been given.

The preceding paragraphs show that both from the side of automata theory and from the side of circuit design, there is a common need for a structural theory by which soliton circuits and soliton automata can be analyzed. This thesis is motivated by the above recognition and its goal is to provide a detailed structural description of soliton graphs and soliton automata. The impacts of our results on the practical design will also be outlined by giving the algorithmic consequences of the theory.

The structural analysis will be carried out on the basis of matching theory. The connection between matching theory and soliton automata was recognized by M. Bartha and E. Gombás in [8]. In this work the concept of perfect internal matchings was introduced, by which we mean matchings covering all the internal vertices in the graph. It is important to emphasize that a perfect internal matching does not necessarily cover the external vertices, which is a significant difference compared to perfect matchings. The authors of [8] also proved the exact counterpart of the Gallai-Edmonds Structure Theorem (cf. [53], [54], [40]) and that of Tutte's theorem (cf. [97]). As a further development of the theory, a more sophisticated description of graphs with perfect internal matchings was worked out in [9], and an algebraic approach to study open graphs and perfect internal matchings has been outlined in [7]. These results provided the foundation of studying soliton automata on the principle of graph matchings.

Using the new concept a soliton graph is defined as an open graph (graph with at least one external vertex) having a perfect internal matching. The edges belonging to a given perfect internal matching correspond to the double bonds in an appropriate state of the molecule-chain, by which it is justified to use the synonym "state" for these matchings. The states of the soliton automaton associated with such a graph G are the states of G, the input alphabet consists of the ordered pair of external vertices, and the state transitions are induced by making alternating walks between the external vertices given by the input. The effect of such a walk is that the status of each edge is exchanged dynamically step by step while making a walk, and by the time the walk is finished a new state is reached. This approach models the effect of a soliton wave travelling along the molecule network forming the basis of a soliton circuit.

Applying matching theory for the analysis of chemical compounds is not a new idea; it has been known for a long time that certain properties of these structures can be studied effectively through the topological model of the molecule. In particular, organic chemists have begun to study the graphs of "conjugated" compounds, that is, compounds the molecules of which possess an alternating pattern of single and double bonds. The name *Hückel graphs* was given to the graphs of such compounds. Hückel graphs play an important role in the area called "resonance theory". For a survey see [96].

The analysis of soliton automata and soliton circuits is not the only reason for the study of perfect internal matchings. We can obtain a common generalization of perfect matchings and perfect internal matchings in the following way.

Let S be a given subset of the vertices of a graph G. Then a matching is called *perfect S-matching* if it covers all vertices of S, but the vertices not contained in S – terminal vertices – are not expected to be covered. Considering a conjugated system where some designated atoms are not expected to join by a double bond in each state of the system, – let us call it open conjugated system – then it is easy to notice the analogy to S-matchings. In other words, as a generalization of soliton automata, we can define automata associated with open conjugated systems in a way that the underlying object is a graph having a perfect S-matching, the set of states is equal to the set the perfect S-matchings, the input symbols are the ordered pair of terminal vertices, and transitions are realized by alternating walks connecting the input pair in an analogous fashion to soliton walks. By a simple procedure ([79]) each graph with a perfect S-matching can be transformed to a soliton graph such that the structure of the resulted graph will be equivalent to the structure of the original graph both from matching-theoretic and automata-theoretic point of view. In other words, any theorem stated for perfect internal matchings can be adapted for perfect S-matchings, and for any automata associated with an open conjugated system there exists an isomorphic soliton automaton. By the above facts, any result of this thesis concerning perfect internal matchings (soliton automata) can be also formulated for perfect S-matchings (respectively, for automata associated with open conjugated systems).

The subject of this thesis is to give a structural analysis of soliton graphs and soliton automata. Our results can be summarized as follows.

(1) We give Tutte type characterizations of soliton graphs by proving two structural theorems on maximal splitters which are the generalizations of maximal barriers.

(2) We generalize the canonical partition of elementary graphs for all graphs having a perfect internal matching.

(3) We work out a decomposition of soliton graphs into elementary components, and these components are given a structure reflecting the order in which they can be reached by external alternating paths. We prove that, based on this structure, the set of elementary components can be grouped pairwise disjoint families. Furthermore, a partial order is established among the families, which reflects the order in which they are reached by external alternating paths.

(4) We characterize the subgraphs determined by the families, and with the help of this result, a linear time algorithm is provided to isolate the families.

(5) We define the Automaton Construction Problem (ACP) for soliton graphs and give a matching-theoretic characterization of soliton transitions which leads an algorithmic solution of ACP.

(6) We give a complete characterization of soliton automata with a single external vertex.

(7) An elementary decomposition of soliton automata is worked out with respect to special type of α_0^{ε}-product. The class of soliton automata is characterized by elementary soliton automata and full automata.

(8) We define the Automaton Description Problem (ADP) for soliton graphs, and work out a structure encoding of soliton graphs which is equivalent to the original graph concerning ADP.

(9) The class of deterministic soliton automata is characterized by generalized trees and chestnuts. The class of partially deterministic soliton automata is characterized by generalized trees and full automata. A polynomial time algorithm is given which decides if a soliton graph is deterministic.

The thesis consists of six chapters, of which the contents are the following.

The opening chapter presents the common preparatory notion, notation and terminology.

In Chapter 2, we review the preliminary results concerning soliton automata. The original definition was based on a weighted graph model. Here we present a model on the principle of graph matchings, and proves its equivalence to the original concept.

Then we present our results in detail as follows.

In Chapter 3, we introduce the concept of splitters in soliton graphs, by which we mean a set of internal vertices such that if its two elements are connected by an extra edge, then this edge is not allowed, i.e. not contained in any state of the graph. The concept of factor-critical graphs is also generalized, and new characterizations are given for such graphs. We will prove two Tutte type theorems on splitters. The first theorem states that for any maximal splitter X, the difference between the cardinality of X and the number of odd components of $G - X$ containing internal vertices only is at most 1. In the second Tutte type theorem we prove that any maximal inaccessible splitter X is a barrier, i.e. the cardinality of X is equal to the number of connected components of $G - X$ consisting of an odd number of internal vertices.

In Chapter 4, after proving some technical lemmas, we generalize the canonical partition of elementary graphs (the allowed edges of which form a connected spanning subgraph) for all graphs having a perfect internal matching. This partition is given by the sets obtained as the restriction of a splitter to an elementary component. Based on this partition the elementary components are given a structure reflecting the order in which they can be reached by external alternating paths. Using this structure it is shown that the set of elementary components can be grouped into pairwise disjoint families. We prove that any external alternating path can enter a family through a distinguished canonical class only. The families themselves are arranged in a partial order $\overset{*}{\mapsto}$ according to the order they can be covered by external alternating paths. The location of impervious edges (edges not traversed by any external alternating trail) in the structure is also characterized. Finally we characterize graphs containing a unique non-degenerate family, and using this result we develop a linear-time algorithm isolating the families.

In Chapter 5, we define the Automaton Construction Problem (ACP) for soliton graphs and give a matching-theoretic characterization of soliton transitions leading an algorithmic solution of ACP. Then, generalizing a result of Dassow and Jürgensen, we characterize soliton automata with a single external vertex by showing that these automata are either full (there is a transition between any two states) or semi-full (there is transition between any two distinct states, but there is no self-transition). As a special type of α_0^{ε}-product, we introduce the concept of canonical product. Based on the elementary decomposition of soliton graphs, we characterize the class of soliton automata as the class of automata obtained by canonical products of elementary soliton automata and full automata. Finally we investigate the Automaton Description Problem (ADP) for soliton graphs both from descriptional and computational point of view. We work out the so-called Elementary Structure Encoding by which each soliton automaton has a code with better descriptional complexity for ADP. An efficient method is also given for the construction of the structure code of any soliton graph.

In Chapter 6, we give a matching-theoretic characterization of deterministic soliton graphs, i.e. we prove that a graph is deterministic iff each connected component of its viable part is either a chestnut (a graph consisting of an even cycle and a few trees such that any two vertices are at even distance from each other) or it does not contain an alternating cycle of even length. The concept of partially deterministic soliton automata is also defined, by which we mean that each of its external elementary components is deterministic. For the main result of the chapter we introduce a reduction method for soliton graphs, which preserves isomorphism. We will prove that an elementary graph is deterministic iff it can be reduced to a graph without even-length cycle, called a generalized tree. We characterize the class of deterministic soliton automata as the class of automata obtained by disjoint products (special quasi-direct ε-products) of generalized trees and chestnuts. The class of partially deterministic soliton automata is characterized as the class of automata obtained by canonical products of generalized trees and full automata. Furthermore, we present an $\mathcal{O}(n^3)$ time algorithm which decides for any graph if it is a deterministic soliton graph.

Finally, we summarize the results of the thesis and mention some open problems.

This thesis is strongly based on the papers [11], [15], [17], [18], [19], and [78].

Chapter 1

Basic notions and notations

1.1 Sets, relations and algebraic structures

In this section we recall the necessary notions and notations concerning sets, relations, functions and algebraic structures.

Sets will generally be denoted by upper-case Latin letters with or without indices, and their elements by the corresponding lower case Latin letters.

For a set A, $|A|$ denotes the *cardinality* of A, and we write 2^A for the *power set* of A, i.e. the set of all subsets of A. The notation $a \in A$ means that a is an *element* of A; the opposite case is expressed by $a \notin A$. The *empty set* is denoted by $\emptyset$.

Given two sets A and B, $A \subseteq B$ means that A is a *subset* of B, $A \subset B$ stands for that A is a *proper subset* of B, and $A \nsubseteq B$ denotes that A is *not a subset* of B. Moreover, $A \setminus B$ denotes the *difference* of A and B, i.e. $A \setminus B$ consists of all elements of A which are not in B.

Let $(A_i \mid i \in I)$ be a family of subsets of a certain set indexed by the elements of set I. Then $\bigcap(A_i \mid i \in I)$ stands for their *intersection* , and $\bigcup(A_i \mid i \in I)$ is their *union* . If I is finite, say $I = \{1, \ldots, k\}$, then we frequently write $A_1 \cap \ldots \cap A_k$ (or $\bigcap_{i=1}^{k} A_i$) and $A_1 \cup \ldots \cup A_k$ (or $\bigcup_{i=1}^{k} A_i$) for $\bigcap(A_i \mid i \in I)$ and $\bigcup(A_i \mid i \in I)$, respectively. If for any distinct $i, j \in I$, $A_i \cap A_j = \emptyset$, then we say that the sets $(A_i \mid i \in I)$ are *pairwise disjoint*,

We sometimes define a set A as the collection of all elements a satisfying certain properties $P_1, \ldots, P_k$. For such A, we use the notation

$$A = \{a \mid a \text{ satisfies } P_1, \ldots, a \text{ satisfies } P_k \}.$$

In the sequel N will stand for the set of all positive integers and let $N_0 = N \cup \{0\}$. Further, for every integer $k \in N$, $[k]$ denotes the set $\{1, \ldots, k\}$.

For sets $A_1, \ldots, A_k$ $(k \in N)$, let $A = A_1 \times \ldots \times A_k$ denote the set

$$A = \{(a_1, \ldots, a_k) \mid a_1 \in A_1, \ldots, a_k \in A_k\},$$

the *Cartesian product* of $A_1, \ldots, A_k$. Moreover, if $A_i = B$ for each i $(i = 1, \ldots, k)$, then A is also called the k^{th} *Cartesian power* of B; in notation $A = B^k$.

Let $(A_i \mid i \in I)$ be a collection of sets indexed by the elements of set I. Then their *disjoint union* is defined as $\bigsqcup(A_i \mid i \in I) = \bigcup(A_i \times \{i\} \mid i \in I)$. Normally $A_i \times \{i\}$ is identified with A_i in an obvious way and the element i is not mentioned; it serves only as a "tag" to make the sets pairwise disjoint. Again, if I is finite, say $I = \{1, \ldots, k\}$, then generally we write $A_1 \sqcup \ldots \sqcup A_k$ for $\bigsqcup(A_i \mid i \in I)$.

Given two sets A_1, A_2, a subset $\tau \subseteq A_1 \times A_2$ is a (binary) *relation* from A_1 to A_2. We also write $a_1 \tau a_2$ instead of $(a_1, a_2) \in \tau$. The opposite case will be expressed by $a_1 \not\tau a_2$.

If $\tau \subseteq A_1 \times A_2$ is a relation, then its *inverse* is the relation τ^{-1} from A_2 to A_1, defined by

$$\tau^{-1} = \{(a_2, a_1) | (a_1, a_2) \in \tau)\}.$$

Take two relations $\tau_1 \subseteq A_1 \times A_2$ and $\tau_2 \subseteq A_2 \times A_3$. The *composition* of τ_1 and τ_2 is the relation $\tau_1 \circ \tau_2$ from A_1 to A_3 for which $(a_1, a_3) \in \tau_1 \circ \tau_2$ if and only if there is an $a_2 \in A_2$ with $(a_1, a_2) \in \tau_1$ and $(a_2, a_3) \in \tau_2$.

A relation from A to A is called a *relation on A.*. The *identity relation* on A is $Id(A) = \{(a, a) \mid a \in A\}$. For any relation τ on a set A and nonnegative integer n, we define the *power* τ^n by the following induction: $\tau^0 = Id(A)$ and $\tau^n = \tau \circ \tau^{n-1}$, for $n > 0$.

The relation τ on set A is said to be

(i) *reflexive* if $Id(A) \subseteq \tau$,

(ii) *symmetric* if $\tau^{-1} \subseteq \tau$,

(iii) *antisymmetric* if $\tau \cap \tau^{-1} \subseteq Id(A)$,

(iv) *transitive* if $\tau^2 \subseteq \tau$.

A relation on A which is reflexive, symmetric and transitive is called an *equivalence relation* on A. If τ is an equivalence relation on A, then for every $a \in A$ we set

$$a/\tau = \{b \mid b \in A, (a, b) \in \tau\}.$$

This notation is extended to an arbitrary subset B of A by

$$B/\tau = \{b/\tau \mid b \in B\}.$$

A *partition* of a set A is a set π of pairwise disjoint nonempty subsets A_i $(i \in I)$ such that $\cup(A_i \mid i \in I) = A$. Each A_i $(i \in I)$ is called a *block* of π. It is well-known that, if τ is an equivalence relation on A, then A/τ is a partition of A, and every partition of A can be given in this way.

The *transitive closure* and the *reflexive, transitive closure* of a relation τ on A are the relations $\tau^+ = \cup(\tau^n \mid n \in N)$ and $\tau^* = \cup(\tau^n \mid n \in N_0)$, respectively. Moreover, the *reflexive, symmetric and transitive closure* of τ is $\cup((\tau \cup \tau^{-1})^n \mid n \in N_0)$. Clearly, the reflexive, symmetric and transitive closure of τ is an equivalence relation.

A reflexive, antisymmetric and transitive relation τ on a set A is a *partial order* on A. It is easy to see that the converse of a partial order is also a partial order. A *linear order* on a set A is a partial order τ on A such that $a \tau b$ or $b \tau a$ holds for arbitrary two elements a and b of A.

A *mapping* or *function* from a set A to a set B is a relation $\varphi \subseteq A \times B$ such that, for every $a \in A$, there is exactly one $b \in B$ satisfying $a \varphi b$. If φ is a mapping from A to B, then we usually write $\varphi : A \to B$. The fact that $a \varphi b$ is also expressed by $\varphi(a) = b$. If $\varphi(a) = b$, then b is the *image* of a. We extend the notation $\varphi(a)$ to an arbitrary subset $A' \in A$ by $\varphi(A') = \{b \mid b = \varphi(a)$ for some $a \in A'\}$. A mapping $\varphi : A \to B$ is *onto* if $\varphi(A) = B$, and if in addition, different elements of A have different images, then φ is called a *bijection* .

The *composition* of two mappings $\varphi : A \to B$ and $\psi : B \to C$ is the composition $\varphi \circ \psi$ of φ and ψ as relations. Clearly, $\varphi \circ \psi$ is a mapping from A to C. The *restriction* of a mapping $\varphi : A \to B$ to a subset $A' \subseteq A$ is the mapping $\varphi|_{A'} : A' \to B$ defined by $\varphi|_{A'} = \varphi \cap (A' \times B)$.

A *multi-set* $\mathcal{M}$, over a set A, is a function $\mathcal{M} : A \to N_0$. The nonnegative integer $\mathcal{M}(a) \in N_0$ is the *number of appearances* of the element a in the multi-set $\mathcal{M}$.

Given a nonempty set A, a mapping from A^2 to A is called a *binary operation* on A. A binary operation σ on A is *associative* if for all $a_1, a_2, a_3 \in A$, $\sigma(a_1, \sigma(a_2, a_3)) = \sigma(\sigma(a_1, a_2), a_3)$

holds. An element $e \in A$ is called *identity element* for σ if $\sigma(a,e) = a = \sigma(e,a)$ with all $a \in A$. Note that the identity element is uniquely determined for any binary operation.

The pair (A,σ), where A is a nonempty set and σ is an associative binary operation on A, is termed a *monoid* if there exists the identity element e for σ. If, in addition, for every $a \in A$ there is an $a^{-1} \in A$ such that $\sigma(a,a^{-1}) = e = \sigma(a^{-1},a)$, then (A,σ) is a *group*. It is easy to prove that for every $a \in A$ there is exactly one a^{-1} satisfying the above equalities, which is called the *inverse* of a. Furthermore, an element a is said to be an *involutorial element* if $\sigma(a,a) = e$. *Trivial* groups are those which consist of a single element.

If (A,σ) is a group and $(B,\sigma|_{B^2})$ with $B \subseteq A$ is also a group, then $(B,\sigma|_{B^2})$ is called a *subgroup* of (A,σ). We say that (A,σ) is *generated* by a subset $C \subseteq A$ if there is no a proper subset B of A for which $C \subset B$ and $(B,\sigma|_{B^2})$ is a group.

Let $(A_1,\sigma_1),\ldots,(A_k,\sigma_k)$ with $k \in N$ be monoids. The *direct product* of $(A_1,\sigma_1),\ldots,(A_k,\sigma_k)$ is the pair (A,σ), where $A = A_1 \times \ldots \times A_k$ and σ is a binary operation on A such that

$$\sigma((a_1^1,\ldots,a_k^1),(a_1^2,\ldots,a_k^2)) = (\sigma_1(a_1^1,a_1^2),\ldots,\sigma_k(a_k^1,a_k^2))$$

for all $a_i^j \in A_i$ with $i \in [k], j \in [2]$. It is easy to see that the above construction gives a monoid such that if each (A_i,σ_i), $i \in [k]$ is a group, then (A,σ) is also a group.

A *permutation* is a mapping from a finite set onto itself. It is well known that the set of all permutations of a given set Ω together with the operation of compositions of functions constitute a group, the *full permutation group* on Ω. Each subgroup of this full permutation group is a *permutation group* on Ω. We call a permutation group $\mathcal{G}$ on Ω *transitive* if for any $\alpha,\beta \in \Omega$, there is some $g \in \mathcal{G}$ such that $g(\alpha) = \beta$. The *symmetric group* of order n – denoted by $\mathcal{S}_n$ – is the group of permutations of the set $[n]$.

1.2 Algorithms and computational complexity

An *algorithm* consists of a set of valid inputs and a sequence of instructions each of which can be composed of elementary steps, such that for each valid input the computation of the algorithm is uniquely defined finite series of elementary steps which produces a certain output. Generally the complexity of an algorithm is measured by its running time. In analyzing running times we ignore constant factors. This not only simplifies the analysis, but it is also realistic from practical point of view: For large enough problem sizes the relative efficiency of an algorithm – given by the running time as an asymptotic function of input size – does not depend on the constant factors. Moreover, this simplification allows us to ignore details of the machine model, thus giving us a complexity measure that is independent from the choice of the computational model used. Note that any machine model used for this goal is deterministic (we will see later that it is important from complexity theoretic point of view). The historically first computational model is the (deterministic) Turing-machine, introduced in the pioneer paper ([95]). Because of technical reasons, i.e. space restrictions, we omit the formal definition of the Turing machine, the reader not familiar with the concept should consult with [89].

We shall measure the running time of an algorithm as an asymptotic function of the worst-case input data. The following notation will be used for this goal: If φ and ψ are functions from $X \subseteq N_0^k$ $(k \in N)$ into the set of positive real numbers, we say that "φ is $\mathcal{O}(\psi)$" (and sometimes write $\varphi = \mathcal{O}(\psi)$) if there are positive constants c_1 and c_2 such that $\varphi(x) \leq c_1 \cdot \psi(x) + c_2$ for all $x \in X$. Therefore if A is an algorithm which accepts inputs from a set X, and $g = \mathcal{O}(f)$ for the function g defined by the relation $\{(x, g(x) \mid x \in X,\ g(x)$ is the number of elementary steps of A on x $\}$ with some function f, then we say that *A runs in $\mathcal{O}(f)$ time.* . We also say that the *running time* (or the *time complexity*) of A is $\mathcal{O}(f)$.

The input to any algorithm of this thesis consists of a list of integers. The *input size* of an instance with integer data is the total number of bits needed for its binary representation. An algorithm is said to run in *polynomial time* if there is an integer k such that it runs in $\mathcal{O}(n^k)$ time, where n is the input size, and all numbers in intermediate computations can be stored with $\mathcal{O}(n^k)$ bits. Furthermore, if $k = 1$ in the above definition, then we say that the running time of the given algorithm is *linear*.

Generally, by an *efficient algorithm* we mean one whose running time is polynomial. We say that a problem defined by the function $f : X \to Y$ is *computable in polynomial time* if there is an algorithm A running in polynomial time which computes f; i.e. the input set of A is X and it computes $f(x)$ for each input $x \in X$. The class of polynomially computable problems is denoted by P.

A larger class of problems, the class NP also plays an important role in complexity theory. We say that a problem is in NP if it can be computed in polynomial time on a nondeterministic Turing machine. We note that the computational power of deterministic and nondeterministic Turing machines are the same, but it is not known if they are equivalent from the point of view of polynomial computability, i.e. it is an open question whether P=NP. In the line of this research the so called *NP-hard problems* play a central role. These problems are those that are hardest in the sense that if one has a polynomial-time algorithm then so does every problem in NP. If, in addition, the given problem belongs to NP, then it is called *NP-complete* . Again, for the formal definitions of the above notions see [89].

1.3 Graphs

Here we present a collection of basic graph theoretic concepts to be used throughout the thesis. Our notation and terminology will be compatible with that of [85], except that the words "point" and "line" will be replaced by "vertex" and "edge", respectively.

An *undirected graph* (or simply *graph*) G consists of a finite set of elements $V(G)$ called *vertices* and a multi-set of unordered pairs of vertices $E(G)$ called *edges* . Note that we allow "multiple" or "parallel" edges. When multiple edges are not allowed, we shall call the corresponding graph *simple*. Also, in a graph G we will allow *loops*, i.e. edges of the form (v, v) with $v \in V(G)$, unless otherwise specified. Graphs without loops will be referred to as *loop-free* graphs.

If $e = (v_1, v_2)$ is an edge in graph G, then we say that e *connects* vertices v_1 and v_2, e is *incident* with v_1 and v_2, and vertices v_1 and v_2 are *the endpoints* of e. If there exists an edge connecting vertices v and w, then v and w are said to be *adjacent*. Two edges sharing at least one endpoint are also referred to as *adjacent*.

For a graph G, the *degree* of a vertex v, denoted by $d_G(v)$ (or simply by $d(v)$ if G is understood), is the number of occurrences of v as an endpoint of some edge in $E(G)$. According to this definition, every loop around v contributes two occurrences to the count.

In a graph G, a *walk* of *length* n is a sequence $\alpha = v_0, e_1, \ldots, e_n, v_n$, $n \geq 0$, of alternating vertices and edges. This sequence indicates the starting point $v_0 \in V(G)$ of α and the vertex v_j, $j \in [n]$, that α reached after traversing the j-th edge e_j. Then v_0 and v_n are referred to as the *endpoints* of α. The notation $\alpha[v_i, v_j]$ with $1 \leq i \leq j \leq n$ will be used for the *subwalk* of α between v_i and v_j, i.e., $\alpha[v_i, v_j] = v_i, e_{i+1}, \ldots, e_j, v_j$. For any walk α, the notation α^{-1} will be used to represent the reverse of α. For technical reasons we shall assign an orientation (e.g. clockwise/anticlockwise) to all loops occurring in walks. By a *backtrack* in a walk we mean the traversal of the same edge twice in a consecutive way. However,as the only exception, the traversal of a loop in the above way is not considered to be a backtrack. (In such cases it is supposed that once a loop has been traversed in one direction, it must be traversed in the same direction immediately afterwards.) If all edges in a walk are distinct, the walk is called a *trail*, and if, in addition, the vertices are also distinct, the trail is a *path*. The concepts of *subtrail* and *subpath* is defined in an analogous fashion to subwalk. Finally a *cycle* is a trail which can be decomposed into a path and an edge connecting the first and the last vertex of the path. Note, that a loop is also a (trivial) cycle according to the above definition.

A *subgraph* G' of G is a graph such that $V(G') \subseteq V(G)$ and $E(G') \subseteq E(G)$. If G' is a subgraph of G such that every edge connecting two vertices of G' which lies in G also lies in G', we call G' an *induced* subgraph of G. If $X \subseteq V(G)$ then the subgraph of G *induced by* X – in notation $G[X]$, or just $[X]$ if G is understood – is the induced subgraph of G for which $V(G[X]) = X$. The notation $G - X$ is a shorthand for $G[V(G) \setminus X]$.

We say that graphs G and G' are *isomorphic* if there exist bijections $\phi : V(G) \to V(G')$ and $\psi : E(G) \to E(G')$ such that for any vertex $v \in V(G)$ and for any edge $e \in E(G)$, e is incident with v if and only if $\psi(e)$ is incident with $\phi(v)$.

A set of vertices or edges S in a graph is said to be *minimal (maximal)* with respect to property P, if the set has property P, but there does not exist a set S' with property P such that S' is a proper subset of S (S is a proper subset of S', respectively). The terms *minimal* and *maximal* subgraphs are defined analogously.

A graph is *connected* if every two vertices are connected by a path. A maximal connected subgraph of G is referred to as a *connected component* of G. Connected components are *even* or *odd* according to whether their vertex sets have even or odd cardinality. A *trivial* connected component is one that consists of a single edge .

If the vertex set of a graph G can be partitioned into two disjoint non-empty sets, $V(G) = A \cup B$, such that all edges of G connect a vertex of A to a vertex of B, we call G *bipartite* and refer to $A \cup B$ as the *bipartition* of G. It is easy to prove that a graph is bipartite if and only if it does not contain a cycle of odd length. A graph G containing no cycles is called a *forest* and if, in addition, G is connected, it is called a *tree*. The *distance* between vertices v and w in a tree is the length of the unique path connecting them. A *spanning tree* T of a connected graph G is a subgraph of G such that $V(T) = V(G)$ and T is a tree.

A *rooted tree* T is a tree with a distinguished vertex r, called the *root*. If v and w are distinct vertices such that v is on the path from r to w, v is an *ancestor* of w and w is a *descendant* of v. If v is an ancestor of w and v and w are adjacent, then v is called the *parent* of w and w is a child of v. A vertex with no children is a *leaf*. The *depth* of a vertex v in a rooted tree T is defined as the distance from v to the root. The *subtree* rooted at vertex v is the rooted tree consisting of the subgraph induced by the descendants of v, with root v. The *nearest common ancestor* of two vertices v and w is the deepest vertex that is an ancestor of both.

The *tree traversal* is the process of visiting each of the vertices in a rooted tree exactly once. The idea of a tree traversal can be extended to graphs. If G is a graph and s is an arbitrary *start vertex*, we carry out a *search* of G starting from s by visiting s and then repeating the following step until there is no unexamined edge (v, w) such that v has been visited:

Search step: Select an unexamined edge (v, w) such that v has been visited and examine it, visiting w if w is unvisited.

Such a search visits each vertex reachable from s exactly once and examines exactly once each edge (v, w) such that v is reachable from s. The search also generates a spanning tree of the subgraph induced by the vertices reachable from s, defined by the set of edges (v, w) such that examination of (v, w) causes w to be visited.

The order of edge examination defines the kind of search. In a *depth-first search*, we always select an edge (v, w) such that v was visited most recently. In a *breadth-first search*, we always select an edge (v, w) such that v was visited least recently. Both depth-first search and breadth-first searches takes $\mathcal{O}(|E(G)|)$ time if implemented properly (see eg. [92] or [94]).

If the edges of a graph have a direction assigned to them, we have what is known as a "directed graph". More precisely, a *directed graph*, or *digraph*, D consists of a set of *vertices* $V(D)$ and a set of ordered pairs of vertices $E(D)$ called *edges*. The number of edges having v as their second vertices is called the *indegree* of v and is denoted by $d^-(v)$. Similarly, the *outdegree* of vertex v is the number of edges having v as their first vertex and is written $d^+(v)$. The definitions of walk, trail, path and cycle must be modified somewhat in the case of directed graphs. In each of these alternating sequences of vertices and edges, we shall insist that each (directed) edge connect the vertex before it to the vertex after it in the sequence. An *acyclic* digraph is one containing no (directed) cycles. A digraph is *strongly connected* if given every ordered pair of vertices (v, w), there is a (directed) path from v to w. A maximal strongly connected subgraph of a digraph D is called a *strong component* of D. It is well-known that if we shrink each strong component of a digraph to a single vertex, then we obtain an acyclic graph. Moreover, the direction of the edges in this shrunken graph determines a partial order among the strong components.

Finally we introduce the concept of (edge) weigthed graphs. To this end let G be an undirected graph, and φ be a mapping from $E(G)$ into N. Then the pair (G, φ) is called a *weigthed graph* with *weight function* φ. Note that more general types of weight functions could be considered; however, for this thesis the notion as introduced is general enough.

1.4 Finite automata

This section contains our notions and notations for finite automata. First, we list some general concepts.

An *alphabet* X is a finite, nonempty set of symbols. A *word* over X is a finite sequence of elements of X. The number of occurrences of symbols in a word w is the *length* of w. For the *empty word*, i.e for the word of length 0, we use the notation ε. Denote by X^* the set of all words over X. In X^* we introduce the *concatenation* of words, i.e., for arbitrary two words $u = x_1 \dots x_m$ and $w = x_{m+1} \dots x_n$ $(x_i \in X_i, i \in [n])$,

$$uw = x_1 \dots x_m x_{m+1} \dots x_n.$$

A *non-deterministic finite automaton* is a triple $\mathcal{A} = (S, X, \delta)$, where S is a non-empty finite set, the *set of states*, X is an alphabet, the *input alphabet*, and $\delta : S \times X \to 2^S$ is the *transition function*. Generally we just use the term "automaton" to mean "non-deterministic finite automaton". An automaton $\mathcal{A} = (S, X, \delta)$ is *deterministic* if for each $s \in S$ and $x \in X$, $|\delta(s, x)| \leq 1$. Automata without outputs as defined here are also referred to as *semi-automata* in the literature.

Let $\mathcal{A} = (S, X, \delta)$ be an automaton. The function δ can be extended to a mapping of $S \times X^*$ into S by

$$\delta(s, \varepsilon) = \{s\}$$

and

$$\delta(s, wx) = \delta(\delta(s, w), x)$$

for $s \in S$, $w \in X^*$, and $x \in X$. For $w \in X^*$ let δ_w denote the transformation of S which is given by

$$\delta_w(s) = \delta(s, w)$$

for $s \in S$. Let

$T(\mathcal{A}) = \{\delta \mid \delta$ is a mapping from S into S and $\delta = \delta_w$ for some $w \in X^*$ $\}$.

With the usual composition of mappings – denoted by $\circ$ – the pair $(T(\mathcal{A}), \circ)$ is a monoid, the *transition monoid* of $\mathcal{A}$. The automaton $\mathcal{A}' = (S', X, \delta')$ is a *subautomaton* of $\mathcal{A}$ if $S' \subseteq S$ and $\delta' = \delta|_{A' \times X}$.

For $i = 1, 2$, let $\mathcal{A}_i = (S_i, X_i, \delta_i)$ be automata. A *homomorphism* of $\mathcal{A}_1$ into $\mathcal{A}_2$ is a pair $\psi = (\psi_S, \psi_X)$ of mappings $\psi_S : S_1 \to S_2$ and $\psi_X : X_1 \to X_2$ which satisfies the equation

$$\{\psi_S(s') \mid s' \in \delta_1(s, x)\} = \delta_2(\psi_S(s), \psi_X(x))$$

for every $s \in S_1$ and every $x \in X_1$. If there is a homomorphism $\psi = (\psi_S, \psi_X)$ such that ψ_S maps from S_1 onto S_2 and ψ_X maps from X_1 onto X_2, then $\mathcal{A}_2$ is the *homomorphic image* of $\mathcal{A}_1$. The homomorphism ψ is an automaton *isomorphism* between $\mathcal{A}_1$ and $\mathcal{A}_2$ if both ψ_S and ψ_X are bijections. The existence of an isomorphism between $\mathcal{A}_1$ and $\mathcal{A}_2$ is denoted by $\mathcal{A}_1 \cong \mathcal{A}_2$. In this case we also say that $\mathcal{A}_1$ and $\mathcal{A}_2$ are *isomorphic*.

Let $\mathcal{A} = (S, X, \delta)$ be a deterministic automaton. It is a *permutation automaton* if $\delta(s, x) \neq \delta(s', x)$ for all $s, s' \in S$ with $s \neq s'$ and for all $x \in X$. The automaton $\mathcal{A}$ is *commutative* if $\delta(s, xy) = \delta(s, yx)$ for all $s \in S$ and all $x, y \in X$.

In the rest of the section we summarize the notions concerning automata products. Representations of automata with general and α_i-products have been studied first for deterministic automata only (cf [55],[56]). In the course of this research Z. Ésik and J. Virágh (cf. [43]) investigated the deterministic ε-products. However, the concepts of general and α_i-products have been recently extended to nondeterministic automata by F. Gécseg, B. Imreh and M. Ito (see [57] and [68]). Here, following the above terminology, we also generalize the notion of ε-products and α_i^ε-products for the non-deterministic case.

Consider the automata $\mathcal{A}_t = (S_t, X_t, \delta_t)$ $(t \in [k], k \in N)$. Their *(general) ε-product* with alphabet X and feedback function ϕ — notation $\prod_{t=1}^{k} \mathcal{A}_t[X, \phi]$ — is the automaton

$$\mathcal{A} = (S, X, \delta), \text{ where}$$

(a) $S = S_1 \times \ldots \times S_k$

(b) $\phi = (\phi_1, \ldots, \phi_k)$ is a mapping, such that
$\phi_t : S_1 \times \ldots \times S_k \times X \to X_t \cup \{\varepsilon\}$, $(t \in [k])$

(c) $\delta((s_1, \ldots, s_k), x) = \delta_1(s_1, \phi_1(s_1, \ldots, s_k, x)) \times \ldots \times \delta_k(s_k, \phi_k(s_1, \ldots, s_k, x))$
for every $x \in X$, $s_t \in S_t$ $(t \in [k])$

The product is called an *α_i^ε-product* $(i \in N_0)$ if for $t = 1, \ldots, k$, each ϕ_t is independent of its jth component whenever $j \geq t + i$. Moreover, if every ϕ_t $(t \in [k])$ maps to the set X_t, then the general ε-product (α_i^ε-product) $\mathcal{A}$ is referred to as a *general product* (*α_i-product*) of $\mathcal{A}_1 \ldots \mathcal{A}_k$.

In the sequel, for α_i^ε-products, in ϕ_t we shall generally indicate those variables on which it may depend. For instance, if $i = 0$, then we frequently write $\phi_1(x)$ for $\phi_1(s_1, \ldots, s_k, x)$, where $(s_1, \ldots, s_k) \in S$ and $x \in X$.

An important restriction of the above concept is the case when each ϕ_t $(t \in [k])$ depends only on the input signal. Then we speak of a *quasi-direct ε-product* of $\mathcal{A}_1 \ldots \mathcal{A}_k$. Obviously, quasi-direct ε-products are special α_0^ε-products. Figure 1.1a) and b) show, respectively, the general form of the α_1^ε- and α_0^ε-products of $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$.

In an α_i^ε-product the set of indices of the component automata are ordered linearly. If we say that each automaton steered by all those component automata which precede it, and speak of feedback only in those cases when an automaton depends on a component automaton preceding it, then i can be considered the length of feedbacks in an α_i^ε-product.

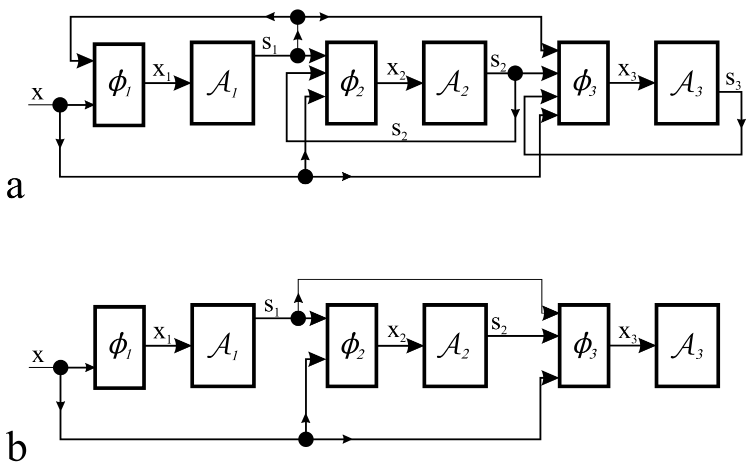

Figure 1.1: General form of the α_i^ε-product with $i \leq 1$.

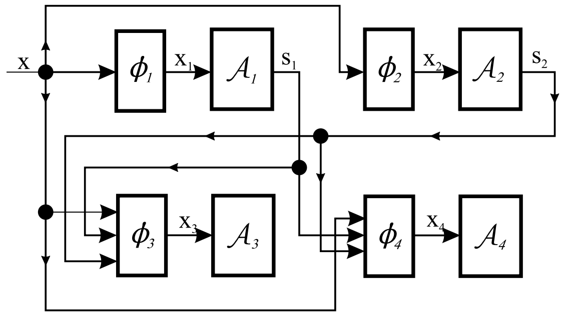

Figure 1.2: General form of the two-level ε-product from $\{\mathcal{A}_1, \mathcal{A}_2\}$ to $\{\mathcal{A}_3, \mathcal{A}_4\}$.

In the thesis a special type of α_0^ε-products will play a central role. In this product the set of component automata are grouped into two "level". The feedback functions of the automata on the first level depend only on the input signal, whereas any automaton on the second level steered by the automata on the first level. (See Figure 1.2). We will call such a product two-level ε-product and give its formal definition below.

Consider the automata $\mathcal{A}_1, \ldots, \mathcal{A}_m$ $(m \in N)$ and let $\mathcal{L} = \{\mathcal{A}_{n+1}, \ldots, \mathcal{A}_m\}$ with $n \leq m$. A *two-level ε-product* from $\mathcal{Q} = \{\mathcal{A}_1, \ldots, \mathcal{A}_n\}$ to $\mathcal{L}$ is an α_0^ε-product $\prod_{j=1}^{m} \mathcal{A}_j[X, \phi]$ such that the following conditions hold for the feedback function $\phi = (\phi_1, \ldots, \phi_m)$.

For each $j \in [m]$,

$\phi_j : X \to X_j \cup \{\varepsilon\}$, if $1 \leq j \leq n$

and

$\phi_j : S_1 \times \ldots \times S_n \times X \to X_j \cup \{\varepsilon\}$, if $n + 1 \leq j \leq m$.

In the followings a set of automata with given conditions will be referred to as a *class of automata*. We say that automata classes $\mathcal{K}$ and $\mathcal{F}$ *coincide up to isomorphism* if for any element of $\mathcal{K}$ there exists an isomorphic automaton of $\mathcal{F}$, and conversely, for any element of $\mathcal{F}$ there exists an isomorphic automaton of $\mathcal{K}$.

Let β be a type of automata products and $\mathcal{K}$, $\mathcal{F}$ be classes of automata. The class $\mathcal{K}$ is said to be *homomorphically complete* for $\mathcal{F}$ with respect to β-products if for any automaton

$\mathcal{A}$ of $\mathcal{F}$ there exist automata $\mathcal{A}_j \in \mathcal{K}$, $j = 1, \ldots n$ such that $\mathcal{A}$ is a homomorphic image of a subautomaton of a β-product of $\mathcal{A}_j$, $j = 1, \ldots, n$. Moreover, the class $\mathcal{K}$ of (deterministic) automata is said to be homomorphically complete with respect to β-products if $\mathcal{F}$ above is the class of all (deterministic) automata. Finally, *isomorphic completeness* is defined in an analogous fashion substituting the terminology "homomorphic image" for "isomorphic image" in the above concepts.

Chapter 2

Soliton automata and matching theory

2.1 Introduction

In this chapter we review the preliminary results concerning soliton automata and develop an equivalent model on the principle of graph matchings. Soliton automata was introduced in [34] in order to capture the logical and computational aspects of the so-called "soliton valves" in polyacetylene chains. The model was defined on the basis of the concept of soliton graphs representing the topological structure of the underlying molecule chains. Soliton graphs were introduced as weighted graphs in which each edge had a weight 2 or weight 1 depending on whether the given edge corresponds to a double or single bond. Therefore a (weighted) soliton graph models a particular state of the molecule chain. A state transition induced by a soliton wave results in a new weighted soliton graph, yet with the same underlying object. The pioneer works by J. Dassow, H. Jürgensen and F. Gécseg dealt with the transformation monoids and the homomorphically complete systems of soliton automata. However, only special cases were analyzed, as the technique based on weighted graphs was not flexible enough to describe the underlying graph structure. Following the above historical way, in Section 2.2 we define the model of soliton automata associated with weighted graphs and present the preliminary results in the topic (cf.[34], [35], [36], [37] and [58]).

The concept that is based on perfect internal matchings introduced in [8] and [9] follows a different approach. It can be shown that each state of the molecule-chain corresponds to a perfect internal matching in the underlying graph, and the necessary arsenal (soliton walks, impervious edges) applied in the analysis of the weighted graph model can be placed into a matching-theoretic framework. This thesis is devoted to the analysis of soliton automata on the principle of matching theory. Nevertheless, the equivalence of the above two concepts has been not proved earlier. Therefore, after reviewing the necessary matching-theoretic background in Section2.3, we build the soliton automaton model on the principle of graph matchings in Section 2.4 with proving its equivalence to the concepts of Section 2.2.

2.2 Definition of the soliton automaton model

In this section, following [34], we introduce the concept of soliton automata as the mathematical model of switching at the molecular level by so-called "soliton valves". Towards this goal, we first define the topological model of the underlying structure, which is a graph representing a molecule chain in which solitons travel along. In this simple model, vertices correspond to the atoms or certain groups of atoms, whereas the edges represent chemical bonds or chains of bonds. The multiplicity of bond (simple or double) is fixed by a weight assignment to the edges. It is assumed that the molecules consist of carbon and hydrogen atoms only, and that among the neighbors of each carbon atom there exists a unique one to which the atom is connected by a double bond.

In order to capture the logical aspects of the process of "soliton valves", we will distinguish three kind of vertices. Vertex v is called *external* if $d(v) = 1$, *internal* if $d(v) > 1$ and *isolated* if $d(v) = 0$. The sets of external and internal vertices of a graph G will be denoted by $Ext(G)$ and $Int(G)$, respectively. An internal vertex adjacent to an external one is called *base*. *External edges* are those that are incident with at least one external vertex, and an *internal edge* is one that is not external. A graph G is called *open* if $Ext(G) \neq \emptyset$, otherwise G is *closed*.

Since in our treatment we are particular about external vertices, we do not want to allow that subgraphs of G possess external vertices other than the ones already in G. Therefore whenever this happens, and an internal vertex v becomes external in a subgraph G' of G, we shall augment G' by a loop around v. This augmentation will be understood automatically throughout the thesis.

Definition 2.2.1 A *weighted soliton graph* is a weighted graph (G, φ) which satisfies the following conditions:

(a) G is a loop-free simple graph;

(b) every connected component of G forms an open graph;

(c) for every $v \in V(G)$, $d(v) \leq 3$;

(d) for every internal vertex v, $\varphi(v) = d(v) + 1$, where $\varphi(v)$ stands for the sum of the weights of all edges incident with v;

(e) if v is an external vertex, then $\varphi(v) \in \{1, 2\}$.

We note that in the original definition (see [34]) the terminology of "soliton graph" was used instead of "weighted soliton graph", but in Section 2.4 we will see that an equivalent unweighted formalization can be given, which makes the above distinction meaningful.

A weighted soliton graph (G, φ) models the "soliton valves" as follows: Each internal vertex v represents a C atom or a C-H group depending on whether $d(v)$ is 3 or 2, respectively. An edge (v, w) of weight $i \in \{1, 2\}$ represents a (CH)-chain with alternating double and single bonds which connects the C atoms of v and w and which begins and ends with an i-fold bond. As the length of such chains does not affect the logic of the model we draw them as length 1 chains; physico-chemical reasons may require different lengths for actual realizations. Finally, external vertices represent the connection to surrounding structures. Figure 2.1 shows an example of a weighted soliton graph and a possible chemical interpretation. Here, and in the examples throughout the section, edges of weight 1 (weight 2) are indicated single lines (double lines, respectively).

For the study of the logical aspects of soliton switching we need to give a graph theoretic formalization of the state transitions induced by soliton waves. Ignoring the physico-chemical details, the effect of a soliton wave propagating along a polycetylene chain is to exchange all single and double bonds. This logical aspect is captured by the concept of soliton walk. Intuitively, a soliton walk is a backtrack-free walk which starts and ends at an external vertex, and alternates on edges with weights 1 and 2. However, the weights of the traversed edges are exchanged dynamically step by step while making the walk.

We note that the terminology of the authors of [34] differs from ours that they call a path what we defined as a walk in Section 1.3.

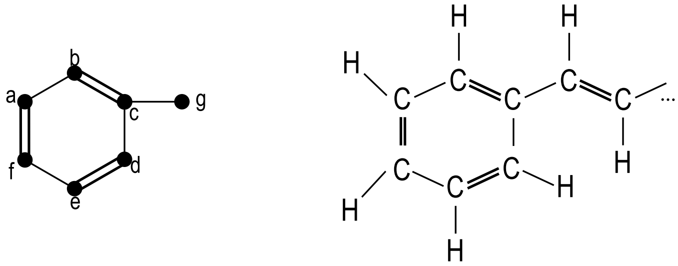

Figure 2.1: A weighted soliton graph with one of its interpretations

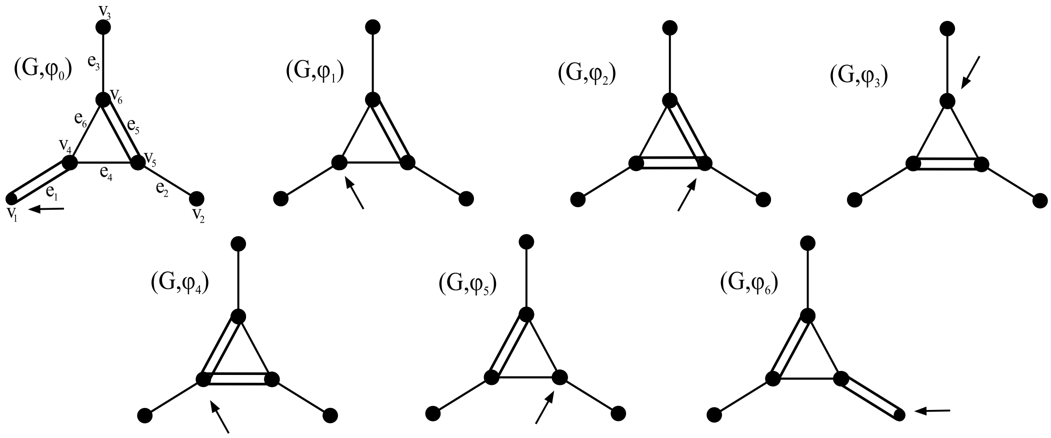

Figure 2.2: A soliton walk with the corresponding sequence of weighted graphs

Before the mathematical definition we give an informal description of soliton walks through the example in Figure 2.2, which illustrates the effects both in the given molecule chain and in its graph model.

(G, φ_0) is the graph representation of the initial state of the system. The walk $\alpha = v_1, e_1, v_4, e_4, v_5, e_5, v_6, e_6, v_4, e_4, e_2, v_2$ corresponding to the given soliton wave results in the sequence $(G, \varphi_1), \ldots, (G, \varphi_6)$ of weighted graphs. In each of them the "position of the soliton" is indicated by an arrow.

The walk starts at vertex v_1 and after traversing edge e_1, the double bond is exchanged for single one; thus the weight of e_1 becomes 1. Then, in each step the weight of the traversed edge is exchanged dynamically. During the walk, if the soliton is about to continue its way on an edge with weight 1 – like in (G, φ_1) at vertex v_4 –, then it might have several alternatives for the next step among the adjacent edges (e.g. in (G, φ_1) both e_4 and e_6 could be chosen). However, if a situation of two adjacent edges with weight 2 occurs – like in graph (G, φ_2) at vertex v_5 –, then the walk must continue on the appropriate edge with weight 2. (Remember, that a soliton walk is backtrack-free.)

Note that the intermediate graphs $(G, \varphi_1), \ldots, (G, \varphi_5)$ are not necessarily weighted soliton graphs. Nevertheless, by the time the walk is finished, a new weighted soliton graph (G, φ_6) is reached.

Definition 2.2.2 Let (G, φ) be a weighted soliton graph. A backtrack-free walk $\alpha = v_0, e_1, v_1, \ldots, v_{n-1}, e_n, v_n$ $(n > 0)$ of G is called a *partial soliton walk* in (G, φ) if the following conditions hold:

(a) v_0 is an external vertex;

(b) $v_1, \ldots, v_{n-1}$ are internal vertices;

(c) there is a sequence $(G, \varphi_0), \ldots, (G, \varphi_n)$ of weighted (not necessarily soliton) graphs that are constructed as follows:

(c1) $\varphi_0 = \varphi$;

(c2) φ_1 is defined in the following way: for all $e \in E(G)$,

$$\varphi_1(e) = \begin{cases} \varphi_0(e), & \text{if } e \neq e_1 \\ 3 - \varphi_0(e_1), & \text{otherwise} \end{cases}$$

(c3) for $i = 1, \ldots, n-1$, the function φ_{i+1} is defined iff φ_i is defined and $\varphi_i(e_{i+1}) - \varphi_i(e_i) = 0$. In this case, for all edges $e \in E(G)$,

$$\varphi_{i+1}(e) = \begin{cases} \varphi_i(e), & \text{if } e \neq e_{i+1} \\ 3 - \varphi_i(e_{i+1}), & \text{otherwise} \end{cases}$$

Such a partial soliton walk is called a *soliton walk* if v_n above is an external vertex.

Given a weighted soliton graph (G, φ) and a partial soliton walk $\alpha = v_0, e_1, v_1, \ldots, v_{n-1}, e_n, v_n$ of (G, φ), let φ_α denote the weight function on $E(G)$ obtained as $\varphi_\alpha = \varphi_n$ according to the construction of Definition 2.2.2 for α. The following lemma from [34] justifies that the introduced concepts make sense.

Lemma 2.2.3 *Let (G, φ) be a weighted soliton graph and α be a soliton walk in (G, φ). Then (G, φ_α) is also a weighted soliton graph.*

Making use of the above result, for a weighted soliton graph (G, φ) and external vertices $v, w \in Ext(G)$, let $\mathcal{S}_G(\varphi, v, w)$ denote the following set of weighted soliton graphs.

$\mathcal{S}_G(\varphi, v, w) = \{(G, \varphi_\alpha) \mid \alpha$ is a soliton walk in (G, φ) which starts at v and ends at w $\}$

Now for a weighted soliton graph (G, φ) consider the sequence

$$\mathcal{G}_0(\varphi), \mathcal{G}_1(\varphi), \mathcal{G}_2(\varphi), \ldots$$

where $\mathcal{G}_0(\varphi) = \{(G, \varphi)\}$, and for $i = 0, 1, 2 \ldots$, the set $\mathcal{G}_{i+1}(\varphi)$ is obtained from $\mathcal{G}_i(\varphi)$ in the following way.

$\mathcal{G}_{i+1}(\varphi) = \{(G, \varphi') \mid (G, \varphi') \in \mathcal{S}_G(\varphi'', v, w)$ for some $(G, \varphi'') \in \mathcal{G}_i(\varphi)$ and $v, w \in Ext(G)\}$.

Then let

$$S(G, \varphi) = \bigcup (\mathcal{G}_i(\varphi) \mid i \in N_0).$$

It is clear that $S(G, \varphi)$ is finite, and can be obtained in finitely many computational steps.

Now making use of the above concepts and notations, we are ready to give the definition of soliton automata.

Definition 2.2.4 The *soliton automaton* associated with underlying weighted soliton graph (G, φ) is defined as the non-deterministic automaton

$$\mathcal{A}(G, \varphi) = (S(G, \varphi), X \times X, \delta)$$

subject to the following conditions:

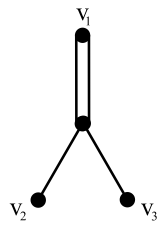

Figure 2.3: Graph (G, φ) of Example 2.2.5

(a) $S(G, \varphi)$ is the set of states;

(b) $X \times X$ is the input alphabet, where $X = Ext(G)$;

(c) $\delta : S(G, \varphi) \times (X \times X) \to 2^{S(G,\varphi)}$ is the transition function with

$$\delta((G, \varphi'), (v, w)) = \begin{cases} \mathcal{S}_G(\varphi', v, w), & \text{if } \mathcal{S}_G(\varphi', v, w) \neq \emptyset \\ (G, \varphi'), & \text{otherwise} \end{cases}$$

for every $(G, \varphi') \in S(G, \varphi)$ and $v, w \in X$.

Observe that the empty walk is not considered a soliton walk. Therefore $\mathcal{S}_G(\varphi, v, w)$ will be nonempty for an external vertex v of G only if there is a (nonempty) soliton walk in (G, φ) starting and ending at v. Otherwise, (v, v) induces the identity transition in (G, φ). In the following we study some examples.

Example 2.2.5 Consider the graph (G, φ) shown in Figure 2.3. One obtains the transitions as shown in Figure 2.4. The resulting automaton has the transition function:

For $i = 1, 2, 3$ and $s \in \{a, b, c\}$,
$\delta(s, (v_i, v_i)) = \{s\}$.

Furthermore,

$$\begin{aligned}
&\delta(a, (v_1, v_2)) = \delta(a, (v_2, v_1)) = \{b\},\\
&\delta(a, (v_1, v_3)) = \delta(a, (v_3, v_1)) = \{c\},\\
&\delta(a, (v_2, v_3)) = \delta(a, (v_3, v_2)) = \{a\},\\
&\delta(b, (v_1, v_2)) = \delta(b, (v_2, v_1)) = \{a\},\\
&\delta(b, (v_2, v_3)) = \delta(b, (v_3, v_2)) = \{c\},\\
&\delta(b, (v_1, v_3)) = \delta(b, (v_3, v_1)) = \{b\},\\
&\delta(c, (v_1, v_3)) = \delta(c, (v_3, v_1)) = \{a\},\\
&\delta(c, (v_2, v_3)) = \delta(c, (v_3, v_2)) = \{b\},\\
&\delta(c, (v_1, v_2)) = \delta(c, (v_2, v_1)) = \{c\}.
\end{aligned}$$

Example 2.2.6 Consider the graph (G, φ) of Figure 2.5. The transitions are represented in Figure 2.6 with the completion of that for $s \in \{a, b, c, d\}$, $i \neq j \in \{1, 2, 3\}$, $\delta(s, (v_i, v_j)) = \delta(s, (v_j, v_i))$ and $\delta(s, (v_i, v_i)) = \{v_i\}$. Note the following fact: The transition in the left column uses the walk $v_1, e_1, v_4, e_6, v_6, e_3, v_3$ from v_1 to v_3, whereas in the right column the walk used is $v_1, e_1, v_4, , e_4, v_5, e_5, v_6, e_3, v_3$. The walks on the diagonal are $v_2, e_2, v_5, e_4, v_4, e_6, v_6, e_3, v_3$ and $v_2, e_2, v_5, e_5, v_6, e_6, e_4, v_5, e_5, v_6, e_3, v_3$.

Example 2.2.7 Consider the graph G in Fig. 2.7. The following weight functions φ_i $(1 \leq i \leq 4)$ define the weighted soliton graph (G, φ_i).

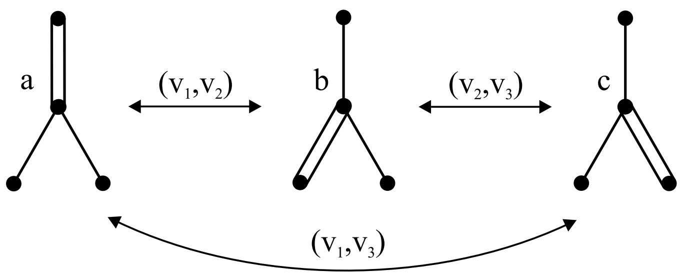

Figure 2.4: Transitions for Example 2.2.5

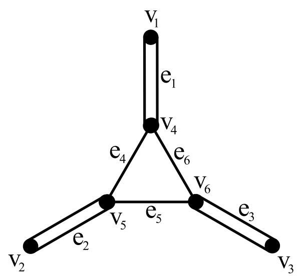

Figure 2.5: Graph (G, φ) of Example 2.2.6

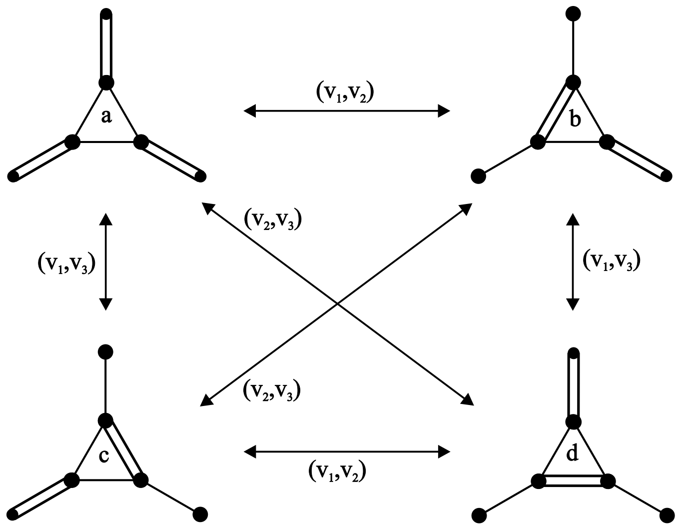

Figure 2.6: Transitions for Example 2.2.6

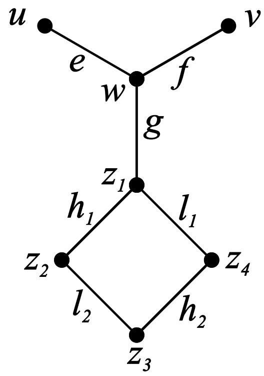

Figure 2.7: The underlying graph of the soliton automaton in Example 2.2.7

For any $x \in E(G)$ let

$$\varphi_1(x) = \begin{cases} 2, & \text{if } x \in \{e, h_1, h_2\} \\ 1, & \text{otherwise} \end{cases}$$

$$\varphi_2(x) = \begin{cases} 2, & \text{if } x \in \{e, l_1, l_2\} \\ 1, & \text{otherwise} \end{cases}$$

$$\varphi_3(x) = \begin{cases} 2, & \text{if } x \in \{f, h_1, h_2\} \\ 1, & \text{otherwise} \end{cases}$$

$$\varphi_4(x) = \begin{cases} 2, & \text{if } x \in \{f, l_1, l_2\} \\ 1, & \text{otherwise} \end{cases}$$

Now the automaton $\mathcal{A}(G, \varphi_1)$ is defined as follows. Let the states $s_h^e = (G, \varphi_1)$, $s_l^e = (G, \varphi_2)$, $s_h^f = (G, \varphi_3)$, and $s_l^f = (G, \varphi_4)$. The transitions of $\mathcal{A}(G, \varphi_1)$ are the following:

$$\begin{array}{l} \delta(s_h^e, (u,v)) = \delta(s_l^e, (u,v)) = \{s_h^f, s_l^f\}, \\ \delta(s_h^f, (v,u)) = \delta(s_l^f, (v,u)) = \{s_h^e, s_l^e\}, \\ \delta(s_h^e, (v,u)) = \{s_h^f\}, \quad \delta(s_l^e, (v,u)) = \{s_l^f\}, \\ \delta(s_h^f, (u,v)) = \{s_h^e\}, \quad \delta(s_l^f, (u,v)) = \{s_l^e\}, \\ \delta(s_h^e, (u,u)) = \{s_l^e\}, \quad \delta(s_l^e, (u,u)) = \{s_h^e\}, \\ \delta(s_h^f, (v,v)) = \{s_l^f\}, \quad \delta(s_l^f, (v,v)) = \{s_h^f\}. \end{array}$$

As an example, the transition $s_h^e \to s_l^f$ on input (u, v) is induced by the soliton walk:

$$uewgz_1h_1z_2l_2z_3h_2z_4l_1z_1gwfv.$$

It is clear that the automata of Examples 2.2.5 and 2.2.6 are deterministic, whereas the automaton of Example 2.2.7 is non-deterministic in the usual sense of the term. However, it also exhibits a different kind of nondeterminism suggested already by the automata in Example 2.2.6: For the same input symbol different walks can be used which, nevertheless, result in the same state transition. This distinction is formalized in the following definition.

Definition 2.2.8 Let (G, φ) be a weighted soliton graph. (G, φ) is called *deterministic* if $|\mathcal{S}_G(\varphi', v, w)| \leq 1$ for all $(G, \varphi') \in S(G, \varphi)$ and all external vertices $v, w \in Ext(G)$. It is called *strongly deterministic* if for every $(G, \varphi') \in S(G, \varphi)$ and for every pair of external vertices $v, w \in Ext(G)$ there is at most one soliton walk from v to w in (G, φ').

Observe that the soliton automaton $\mathcal{A}(G,\varphi)$ associated with a weighted soliton graph (G,φ) is deterministic in the usual automaton theoretic sense if and only if (G,φ) is deterministic. Moreover, we will call a soliton automaton $\mathcal{A}(G,\varphi)$ *strongly deterministic*, if (G,φ) is a strongly deterministic weighted soliton graph.

It is obvious that for any weighted soliton graph (G,φ), the connected components of G act as "independent units". Moreover, it is also clear that *impervious edges*, i.e. edges not contained in any partial soliton walk of any weighted soliton graph $(G,\varphi') \in \mathcal{S}(G,\varphi)$, have no effect on the operation of $\mathcal{A}(G,\varphi)$. Therefore it is useful to consider the following decomposition of weighted soliton graphs.

For a weighted soliton graph (G,φ) let G^+_φ denote the (unweighted) graph obtained from G by deleting the impervious edges of (G,φ). It can be proved (cf. [34]) that $(G^+_\varphi, \varphi|_{E(G^+_\varphi)})$ is also a weighted soliton graph and $\mathcal{A}(G,\varphi) \cong \mathcal{A}(G^+_\varphi, \varphi|_{E(G^+_\varphi)})$. Then the *soliton decomposition* of (G,φ) is given by the weighted soliton graphs $(G_1,\varphi_1),\ldots,(G_k,\varphi_k)$ with $k \geq 1$, where $G_1,\ldots G_k$ are the connected components of G^+_φ and $\varphi_j = \varphi|_{E(G_j)}$ for each $1 \leq j \leq k$. Furthermore, a weighted soliton graph is called *indecomposable* if it is connected and contains no impervious edges.

By summarizing the above facts, in [34] was shown that it is enough to analyze automata associated with indecomposable weighted soliton graphs. Here in Proposition 2.2.11 we reformulate the above result by the concepts of strong isomorphism and disjoint product introduced below, which will be also used several times later in the thesis.

Definition 2.2.9 For $i = 1,2$, let X_i be alphabets and $\mathcal{A}_i = (S_i, X_i \times X_i, \delta_i)$ be automata. We say that $\mathcal{A}_1$ and $\mathcal{A}_2$ are *strongly isomorphic* if there exists a pair $\psi = (\psi_S, \psi_X)$ of bijections $\psi_S : S_1 \to S_2$ and $\psi_X : X_1 \to X_2$ which satisfies the equation

$$\{\psi_S(s') \mid s' \in \delta_1(s,(x,x'))\} = \delta_2(\psi_S(s),(\psi_X(x),\psi_X(x')))$$

for every $s \in S_1$ and every $x, x' \in X_1$.

It is clear that for automata with alphabets such as in Definition 2.2.9, strong isomorphism implies isomorphism. However, the simple example for soliton automata in Figure 2.8 shows that the opposite statement is generally not true. Indeed, it is easy to see that $\mathcal{A}(G_1,\varphi_1)$ and $\mathcal{A}(G_2,\varphi_2)$ are not strongly isomorphic, but $\mathcal{A}(G_1,\varphi_1) \cong \mathcal{A}(G_2,\varphi_2)$ by the following bijections ψ_S and ψ_X.

$\psi_S((G_1,\varphi_1)) = (G_2,\varphi_2)$, $\psi_S((G_1,\varphi_1')) = (G_2,\varphi_2')$,
$\psi_X((v_1,v_1')) = (v_2,v_2)$, $\psi_X((v_1',v_1)) = (v_2',v_2'))$,
$\psi_X((v_1,v_1)) = (v_2,v_2')$, $\psi_X((v_1',v_1')) = (v_2',v_2))$.

The fact that automata classes $\mathcal{K}$ and $\mathcal{F}$ *coincide up to strong isomorphism* is defined in an analogous fashion to isomorphism.

Definition 2.2.10 Let $X_1,\ldots,X_k$ $(k \in N)$ be alphabets and consider the automata $\mathcal{A}_j = (S_j, X_j \times X_j, \delta_j)$ $(j \in [k])$. The *disjoint product* of $\mathcal{A}_1,\ldots,\mathcal{A}_k$ is the quasi-direct ε-product $\prod_{j=1}^{k} \mathcal{A}_j[X \times X, \phi]$ with alphabet $X \times X$ and feedback function $\phi = (\phi_1,\ldots,\phi_k)$ subject to the following conditions:

(a) $X = X_1 \sqcup \ldots \sqcup X_k$.

(b) For each $j \in [k]$ and $x, x' \in X$,

$$\phi_j((x,x')) = \begin{cases} (x,x'), & \text{if } x,x' \in X_j \\ \varepsilon, & \text{otherwise} \end{cases}$$

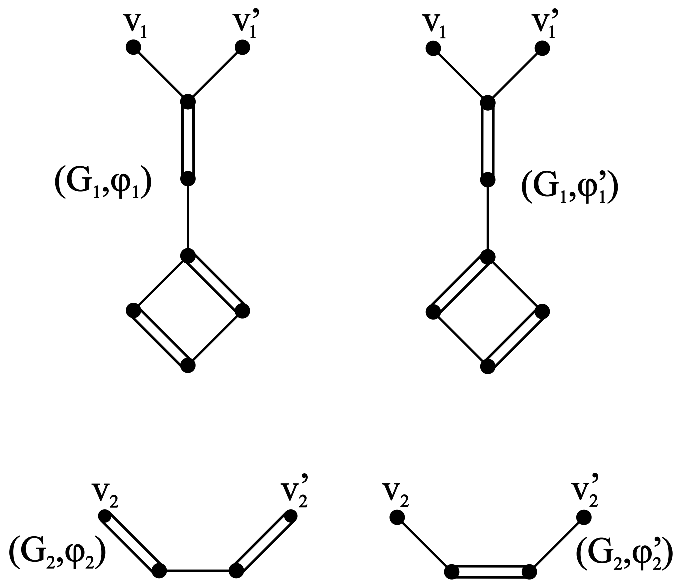

Figure 2.8: Example for isomorphic but not strongly isomorphic soliton automata.

Proposition 2.2.11 *Let* (G, φ) *be a weighted soliton graph and* $(G_1, \varphi_1), \ldots, (G_k, \varphi_k)$ $(k \in N)$ *be the soliton decomposition of* (G, φ)*. Then* $\mathcal{A}(G, \varphi)$ *is strongly isomorphic with the disjoint product of* $\mathcal{A}(G_1, \varphi_1), \ldots, \mathcal{A}(G_k, \varphi_k)$*.*

In the rest of this section we summarize the preliminary results concerning soliton automata by J. Dassow, H. Jürgensen and F. Gécseg (cf. [34, 35, 36, 37, 58]).

First we present the result from [34] characterizing strongly deterministic soliton graphs. For this goal we need the following definition.

Definition 2.2.12 A connected graph G is called a *chestnut* if it has a representation in the form $G = \beta + \alpha_1 + \ldots + \alpha_k$ with $k \geq 1$, where β is a cycle of even length and each α_i $(i \in [k])$ is a tree subject to the following conditions:

(i) $V(\alpha_i) \cap V(\alpha_j) = \emptyset$ for $1 \leq i \neq j \leq k$;

(ii) $V(\alpha_i) \cap V(\beta)$ consists of a unique vertex – denoted by v_i – for each $i \in [k]$;

(iii) v_i and v_j are at even distance on β for any distinct $i, j \in [k]$;

(iv) any vertex $w_i \in V(\alpha_i)$ with $d(w_i) > 2$ is at even distance from v_i in α_i for each $i \in [k]$.

It is easy to check that for any chestnut G, there exists a weight function φ satisfying conditions (d) and (e) of Definition 2.2.1. Therefore, if $d(v) \leq 3$ for any $v \in V(G)$, then (G, φ) is a weighted soliton graph. Figure 2.9 shows an example for a chestnut as a soliton graph.

Theorem 2.2.13 *Let* (G, φ) *be an indecomposable weighted soliton graph. Then* (G, φ) *is strongly deterministic if and only if* G *is a chestnut or a tree.*

The next result (cf. [34]) describes the transition monoids of strongly deterministic soliton automata. For this, we review a few group theoretic terms.

Let $\mathcal{G}$ be a permutation group on a set Ω. A subset Ψ of Ω is called a *block* if for each $g \in \mathcal{G}$ the image $g(\Psi)$ either coincides with Ψ or is disjoint from Ψ. The sets $\emptyset$, $\{\omega\}$, for any

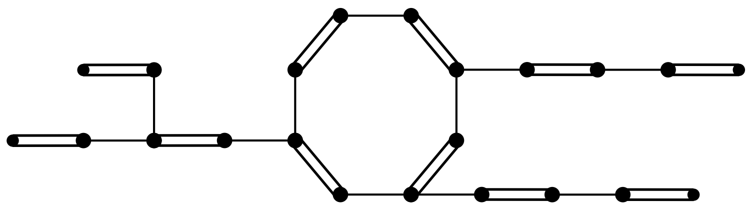

Figure 2.9: A chestnut.

$\omega \in \Omega$, and Ω are the *trivial blocks*. The group $\mathcal{G}$ is called *primitive* if it is transitive and has only trivial blocks.

Theorem 2.2.14 *The transition monoid of a strongly deterministic soliton automaton is a direct product of primitive permutation groups which are generated by involutorial elements.*

As a refinement of the above theorem J. Dassow and H. Jürgensen described in [37] the primitive permutation groups which occur as transition monoids of automata associated with a special class of trees.

Theorem 2.2.15 *Let (T, φ) be a weighted soliton tree such that any two vertices of degree 3 are at even distance from each other. Then the transition monoid of $\mathcal{A}(T, \varphi)$ is a symmetric group.*

The characterization of deterministic soliton automata seems to be far more difficult than that of the strongly deterministic ones. Only two special cases have been analyzed with respect to their transition monoids: deterministic soliton automata with a single external vertex (cf. [35]) and deterministic soliton automata with at most one cycle (cf. [36]). The main result of [35] is stated in Theorem 2.2.16. To state this theorem, we use the concept of *usable cycle*, by which we mean a cycle occuring as a subwalk of some soliton walk.

Theorem 2.2.16 *Let (G, φ) be a deterministic, indecomposable weighted soliton graph with a single external vertex. If (G, φ) contains a usable cycle of even length then G is a chestnut, $\mathcal{A}(G)$ has 2 states and $T(\mathcal{A}(G, \varphi)) \cong \mathcal{S}_2$. Otherwise, $\mathcal{A}(G, \varphi)$ has a single state only and $T(\mathcal{A}(G, \varphi))$ is trivial.*

In order to describe the results of [36] we need some preparation.

Let G be a connected open graph with a single cycle β. Clearly, in this case G has a representation in the form $G = \beta + \alpha_1 + \ldots + \alpha_r$, $r \in N$, such that $\alpha_1, \ldots, \alpha_r$ are pairwise vertex-disjoint trees and for each $i \in [r]$ $V(\alpha_i) \cap V(\beta)$ consists of a single vertex. The above decomposition will be referred to as the *tree-decomposition* of G. Now the main result of [36] can be summarized as follows.

Theorem 2.2.17 *Given an indecomposable weighted soliton graph (G, φ) with a single cycle β and the tree-decomposition of G by $G = \beta + \alpha_1 + \ldots + \alpha_r$. Moreover, let $V_\alpha = \{v_1, \ldots, v_r\}$, where for each $i \in [r]$, v_i denotes the unique common vertex of α_i and of β. Then $T(\mathcal{A}(G, \varphi))$ is a primitive group of permutations if and only if one of the following conditions fails to hold:*

(a) β is an odd-length cycle.

(b) There are three distinct vertices $v_{i_1}, v_{i_2}, v_{i_3}$ of V_α such that α_{i_j} with $j = 1, 2, 3$ consists of a single path and for $s = 1, 2$ the unique odd-length subpath of β connecting v_{i_s} and $v_{i_{s+1}}$ – apart from the endpoints – does not contain vertices of V_α.

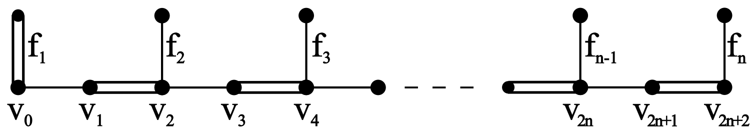

Figure 2.10: The soliton tree (T_n, φ_n)

Finally we will give a summary of [58]. In this paper the connection of the class of deterministic finite automata and that of strongly deterministic soliton automata was investigated with respect to different automata products. In reaching the above goal, the following subclass of strongly deterministic soliton automata is of particular importance.

For $n \in N$ let T_n denote the tree which has a representation in the form $T_n = f_1 + \ldots + f_{n+2} + \alpha$, where $f_1, \ldots, f_{n+2}$ are external edges and $\alpha = v_0, e_1, \ldots, e_{2n+2}, v_{2n+2}$ is a path such that the internal endpoint of f_i $(i \in [n+2])$ is v_{2i-2}. See an example in Figure 2.10.

Now for any $n \in N$, define the weight function φ_n on $E(T_n)$ in the following way (See Fig.2.10 again):

For any $e \in E(T_n)$,

$$\varphi_n(e) = \begin{cases} 2, & \text{if } e = f_1 or e = e_{2j-2} with some 2 \leq j \leq n+2 \\ 1, & \text{otherwise} \end{cases}$$

It is easy to see that (T_n, φ_n) is a weighted soliton tree for any $n \in N$. Then let $\mathcal{G} = \{\mathcal{A}(T_n, \varphi_n) \mid n \in N\}$.

Theorem 2.2.18 *The class $\mathcal{G}$ is homomorphically complete for the class of commutative permutation automata with respect to the α_0-product. Furthermore, $\mathcal{G}$ is homomorphically complete with respect to the α_1-product.*

Note that there is no homomorphically complete class of strongly deterministic soliton automata with respect to the α_0-product. Indeed, it is an immediate consequence of the definition that every strongly deterministic soliton automaton is a permutation automaton (cf. [34]). Moreover, subautomata and homomorphic images of a permutation automaton are also permutation automata, and α_0-products preserve the "permutation property" of automata. (see [58]).

It is known from [42] that the α_2-product is homomorphically equivalent to the general product. Therefore, in studying homomorphic representations by α_i-products of strongly deterministic soliton automata with $i \geq 2$, it is enough to consider general products.

Theorem 2.2.19 *A class $\mathcal{K}$ of strongly deterministic soliton automata is homomorphically complete with respect to the general product (or the α_i-product with $i \geq 2$) if and only if $\mathcal{K}$ contains an automaton whose underlying weighted soliton graph (G, φ) satisfies one of the following three conditions:*

(i) the soliton decomposition of (G, φ) consists of at least two components;

(ii) the soliton decomposition of (G, φ) consists of a single tree;

(iii) the soliton decomposition of (G, φ) consists of a single chestnut with at least two external vertices.

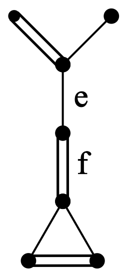

Figure 2.11: Forbidden and mandatory edges in an open graph with perfect internal matching

2.3 Perfect internal matchings in graphs

As we saw in the previous section, some results concerning automata products and transition monoids could be proved for special cases of soliton automata, but the general case seems a complex task. The core of obtaining further results is to describe the graph structure of soliton graphs. The "new" technique by which we will investigate the above problem throughout the dissertation is based on the concept of perfect internal matchings introduced in [9]. To reach this goal we summarize here the necessary notions concerning perfect internal matchings (cf. [9]).

A *matching* M of graph G is a subset of $E(G)$ such that no vertex of G occurs more than once as an endpoint of some edge in M. It is understood that loops are not allowed to participate in M. The endpoints of the edges contained in M are said to be *covered* by M. For a subgraph G' and matching M of G, $M_{(G')}$ will denote the restriction of M to G', i.e. $M_{(G')} = M \cap E(G')$. A matching is called *perfect* if it covers all vertices of G. A *perfect internal matching* of G is one that covers all vertices of $Int(G)$. An edge $e \in E(G)$ is called *allowed* (*mandatory*) if e is contained in some (respectively, all) perfect internal matching(s) of G. *Forbidden edges* are those that are not allowed. We will also use the term *constant edge* to identify an edge that is either forbidden or mandatory. Figure 2.11 shows an example for an open graph G with a perfect internal matching. It is easy to check that edge e is forbidden, whereas edge f is mandatory in G. In this figure, as well as in some of the further ones throughout the thesis, double lines indicate edges that belong to the given matching.

Let G be graph with a perfect internal matching, fixed for the rest of the section, and let M be a perfect internal matching in G. An edge $e \in E(G)$ is said to be M*-positive* (M*-negative*) if $e \in M$ (respectively, $e \notin M$). An *alternating trail* with respect to M (or M-alternating trail, for short) in G is a trail stepping on M-positive and M-negative edges in an alternating fashion. Notice that an alternating trail can return to itself only at its endpoints (the typical cases of maximal external alternating trails can be seen in Figure 2.12). Let us agree that, if the matching M is understood or irrelevant in a particular context, then it will not explicitly be indicated in these terms.

An alternating path is *positive* (*negative*) if it is such at its internal endpoints, meaning that the edges incident with those endpoints are positive (respectively, negative). If both endpoints of an alternating path are external, then it is called a *crossing* (see Figure 2.12a). An *alternating loop* around vertex v is an odd alternating cycle starting from v. Clearly, the first and the last edge of any alternating loop must not be in the given matching. Since we now have a distinct name for odd alternating cycles, we shall reserve the term *"alternating cycle"* for even length ones.

An *external trail (path)* is a trail (path) having an external endpoint. *Internal trails (paths)* are those that are not external. It is clear that any maximal external alternating trail α with

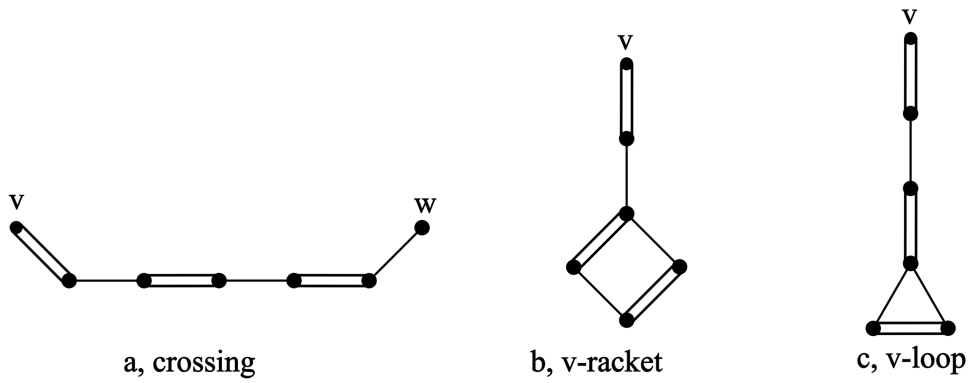

Figure 2.12: Maximal external alternating trails: crossing, v-racket, and v-loop.

a unique external vertex v can be decomposed in the form $\alpha = \alpha_h + \alpha_c$, where α_h, the *handle of* α , is an external alternating path starting from v, whereas α_c, the *cycle of* α, is either an alternating loop or an alternating cycle. With these parameters, α is called an *alternating v-racket* or an *alternating v-loop* depending on whether α_c is an alternating cycle or an alternating loop (see Figure 2.12/b-c). Note that if α_c is an alternating cycle, then α_h is negative, otherwise it is positive.

For any sate M of G, an *M-alternating unit* is either an M-alternating crossing or an M-alternating cycle. *Switching* on an M-positive alternating path or M-alternating unit α amounts to exchanging the status of each edge of α regarding its being present or not being present in M. It is easy to see that the operation of switching on α creates a new matching $S(M, \alpha)$ for G, which matching is a perfect internal matching if the switching is carried out on an alternating unit. An *M-alternating network* is a set of nonempty, pairwise disjoint M-alternating units. Note that, though the alternating units in an alternating network Γ are nonempty, Γ itself can be an empty set. *Switching* on an M-alternating network Γ in M means switching on the units of Γ simultaneously in M. Since the units of Γ do not intersect each other, the resulting perfect internal matching, denoted by $S(M, \Gamma)$, is well-defined. Finally, if G' is a subraph of G, then by an $M_{(G')}$-alternating trail ($M_{(G')}$-alternating network, respectively) we mean one that is entirely contained in graph G'.

We close the section by two results of M. Bartha and E. Gombás ([9]). The first theorem describes the state transitions with the help of alternating networks, while the second one characterizes non-constant edges by alternating units.

Theorem 2.3.1 *For any two perfect internal matchings M_1, M_2 of graph G, there exists a unique mediator alternating network Γ between M_1 and M_2, i.e. $S(M_1, \Gamma) = M_2$ and $S(M_2, \Gamma) = M_1$.*

Theorem 2.3.2 *An edge e of a soliton graph G is not constant if and only if there exists an alternating unit passing through e in every state of G.*

2.4 Soliton automata and perfect internal matchings

Having introduced the necessary matching theoretic concepts, it is easy to establish a correspondence between weight functions with the conditions of Definition 2.2.1 and perfect internal matchings. Indeed, conditions (d) and (e) imply that the weight of each edge in a weighted soliton graph (G, φ) is either 1 or 2, and for every internal vertex v there exists exactly on edge e incident with v such that $\varphi(e) = 2$. Let $M \subseteq E(G)$ consist of those edges which have weight 2. Clearly, M is a perfect internal matching of G. Conversely, every perfect internal matching of G corresponds to a suitable weight function φ satisfying conditions (d) and (e) in Definition 2.2.1. Considering the above facts it is justified to use the name *state* as a synonym for perfect internal matching. The set of states (set of perfect internal matchings) of a graph G will be denoted by $S(G)$.

Based on the above observations we can use the term "equivalent" for a perfect internal matching M of graph G and weight function φ on $E(G)$ if M and φ correspond to each other in the sense of the preceding paragraph. Moreover, it is clear that the conditions (a), (b) and (c) in Definition 2.2.1 impose restrictions on the graph structure only, so that they are irrelevant as far as matchings (or weight functions) are concerned. More exactly, omitting the above conditions makes it possible to use more general techniques and constructions throughout the thesis. Nevertheless, as we will see in Theorem 2.4.10, soliton graphs in generalized sense (soliton graphs without conditions (a), (b) and (c) of Definition 2.2.1) are equivalent to the original concept.

Summarizing the observations of the preceding paragraphs, a *generalized soliton graph G* can be defined as an open graph having a perfect internal matching. In the rest of the section if a generalized soliton graph G is given, and φ is a weight function equivalent to some state of G, then (G, φ) is considered as a generalized weighted soliton graph. In this case all the notions introduced in Section 2.2 are applied for (G, φ) in an analogous way.

In order to place the concept of soliton automata into a matching theoretic framework, first we need to reformulate the definition of (partial) soliton walks. To this end, for a walk $\alpha = v_0, e_1, \ldots, e_n, v_n$, let $n_\alpha(j)$ ($j \in [n]$) denote the number of occurrences of the edge e_j in the prefix $\alpha[v_0, v_j]$.

Definition 2.4.1 A *partial soliton walk* with respect to state M in generalized soliton graph G is a backtrack-free walk $\alpha = v_0, e_1, \ldots, e_n, v_n$ ($n \geq 1$) subject to the following conditions:

(a) v_0 is an external vertex;

(b) for every $j \in [n-1]$, $n_\alpha(j)$ and $n_\alpha(j+1)$ have the same parity if and only if e_j and e_{j+1} are M-alternating, i.e., $e_j \in M$ iff $e_{j+1} \notin M$.

Furthermore, a partial soliton walk is called a *soliton walk* if v_n above is an external vertex.

Making the walk α in state M means creating the edge set $S(M, \alpha)$ by setting for every $e \in E(G)$:

$e \in S(M, \alpha)$ iff $e \in M$ and e occurs an even number of times in α, or $e \notin M$ and e occurs an odd number of times in α.

Note that if α is a crossing or a positive alternating path, then switching on α and making α result in the same matching, by which it is justified to use $S(M, \alpha)$ as a common notation. Moreover, it is easy to see that making the walk does not necessarily result in a perfect internal matching (or even a matching). However, as Proposition 2.4.2 shows, if α is a soliton walk, then $S(M, \alpha)$ is also a state.

Proposition 2.4.2 *Let G be a generalized soliton graph, M be a state of G and α be a soliton walk with respect to M. Then $S(M, \alpha)$ is also a state of G.*

Proof. We will prove a more general statement, from which Proposition 2.4.2 directly follows. Namely, we will show that for any internal vertex w and for any state M of an arbitrary generalized soliton graph, if α is a partial soliton walk with respect to M such that w is not an endpoint of α, then $S(M, \alpha)$ contains a unique edge incident with w and this edge is not a loop.

The proof of the above claim will apply an induction on $c_\alpha(w)$, which denotes, for any internal vertex w and any partial soliton walk α, the number of occurrences of w in α. The basis step, i.e. the statement for vertices w and walks α with $c_\alpha(w) = 0$, is trivial.

For the induction step, let generalized soliton graph G, $M \in S(G)$, $w \in Int(G)$ and partial soliton walk $\alpha = v_0, e_1, \ldots, e_n, v_k$ with respect to M be arbitrary such that $v_0 \in Ext(G)$, $v_k \neq w$ and $c_\alpha(w) > 0$. First we prove the following claim, which guarantees that no loop around w belongs to $S(M, \alpha)$.

(i) Any loop around w is traversed an even number of times by α.

Suppose on the contrary, that e_s is a loop around w with $1 < s < k$ such that $n_\alpha(s)$ is odd and e_s is not traversed by $\alpha[v_s, v_k]$. Construct the graph G_s from G by attaching a new external edge f_s to v_{s-1} and consider the soliton walk $\alpha_s = \alpha[v_0, v_{s-1}] + f_s$ in G_s. It is clear that $c_{\alpha_s}(w) < c_\alpha(w)$ and $f_s \in S(M, \alpha_s)$. Therefore, applying the induction hypothesis and returning to G, we obtain that e_s is the unique edge of $S(M, \alpha[v_0, v_s])$ incident with w. Consequently, for any edge $f \neq e_s$ of G incident with w, either $f \in M$ and f is traversed an odd number of times by $\alpha[v_0, v_s]$, or $f \notin M$ and f is traversed by an even number of times by $\alpha[v_0, v_s]$. However, by Definition 2.4.1, either $e_{s+1} \in M$ and $n_\alpha(s+1)$ is odd, or $e_{s+1} \notin M$ and $n_\alpha(s+1)$ is even. Hence we obtained a contradiction in either case, by which (i) is proved.

For the rest of the proof, starting from v_0, let v_m denote the last occurrence of w in α and let v_l denote the last vertex of $\alpha[v_0, v_m]$ which is different from w. Clearly, if e_m is not a loop, then $l = m - 1$. Now observe that, if we prove the following statement, then we are ready:

(ii) Either of e_{l+1} and e_{m+1} is contained in $S(M, \alpha[v_0, v_l])$.

Indeed, making use of (i) and the induction hypothesis for $S(M, \alpha[v_0, v_l])$, (ii) implies that either, in the case when $e_{l+1} = e_{m+1} \in S(M, \alpha[v_0, v_l])$, e_{l+1} is the unique edge of $S(M, \alpha)$ incident with w, or, in the case when $e_{l+1} \notin S(M, \alpha[v_0, v_l])$ ($e_{m+1} \notin S(M, \alpha[v_0, v_l])$, e_{l+1} (respectively, e_{m+1}) will be the above-mentioned designated edge of $S(M, \alpha)$. We distinguish two cases.

Case (ii/a) Edge e_{l+1} is a loop. Then, applying the induction hypothesis, $n_\alpha(l+1)$ is odd, consequently, either $e_l \in M$ and $n_\alpha(l)$ is odd, or $e_l \notin M$ and $n_\alpha(l)$ is even. It is easy to check that in either case $e_l \in S(M, \alpha[v_0, v_l])$.

Case (ii/b) Edge e_{l+1} is not a loop, i.e. $l + 1 = m$. Then suppose by contradiction that $e_{m-1}, e_m \notin S(M, \alpha[v_0, v_l])$. Clearly, either of e_{m-1} and e_m, say e_{m-1}, is not contained in M – the proof is analogous if $e_m \notin M$ is assumed –, consequently $n_\alpha(m-1)$ is odd. Therefore, either $e_m \in M$ with $n_\alpha(m)$ being odd, or $e_m \notin M$ with $n_\alpha(m)$ being even holds, which is equivalent in both cases to the contradictory fact that $e_m \in S(M, \alpha[v_0, v_l])$.

By the preceding two paragraphs, (ii) is verified, which makes the proof of Proposition 2.4.2 complete. ♢

The following result justifies the use of our terminology by proving the equivalence of Definitions 2.2.2 and 2.4.1.

Proposition 2.4.3 *Let G be a generalized soliton graph, M be a perfect internal matching in G and φ be a weight function on $E(G)$ such that M and φ are equivalent. Then any walk α in G is a partial soliton walk with respect to M if and only if α is a partial soliton walk in (G, φ). Furthermore, if α is a soliton walk, then φ_α and $S(M, \alpha)$ are equivalent.*

Proof. By an induction on the length of α, we will prove a more general statement: α is a partial soliton walk with respect to M iff α is such in (G, φ), and in this case, for each edge $e \in E(G)$, $e \in S(M, \alpha)$ iff $\varphi_\alpha(e) = 2$.

The claim for walks with length 1 is trivial, thus, for the induction step, consider an arbitrary external walk $\alpha = v_0, e_1, \ldots, e_k, v_k$ with $v_0 \in Ext(G)$ and $k > 1$. Obviously, α is a partial soliton walk with respect to M iff the following two conditions hold:

(i) $\alpha' = \alpha[v_0, v_{k-1}]$ is a partial soliton walk with respect to M.

(ii) Either $n_\alpha(k-1)$ and $n_\alpha(k)$ have the same parity with e_{k-1} and e_k being M-alternating, or the parities of $n_\alpha(k-1)$ and of $n_\alpha(k)$ are different with $e_{k-1}, e_k \notin M$.

It is easy to check that – if condition (i) holds – (ii) is equivalent to saying that $e_{k-1} \in S(M, \alpha')$ iff $e_k \in S(M, \alpha')$. Therefore, applying the induction hypothesis with the above observation, (i) and (ii) hold iff α' is a partial soliton walk in (G, φ) and $\varphi_{\alpha'}(e_{k-1}) = \varphi_{\alpha'}(e_k)$. By the above facts, the first part of our statement is clearly proved. Now the proof can be finished by observing that $e_k \in S(M, \alpha)$ iff $e_k \notin S(M, \alpha')$, and for any edge $e \neq e_k$, $e \in S(M, \alpha)$ iff $e \in S(M, \alpha')$. $\diamondsuit$

Our goal now is to make the definition of soliton automata matching invariant. To this end recall from Section 2.2 that impervious edges have no effect on the operations of soliton automata. Because of the above fact we will give the "new" definition of soliton automata with the help of soliton graphs having no impervious edges. In order to reach this goal first we show by the following result from [11] that "impervious property" can be also expressed by external alternating trails.

Proposition 2.4.4 *Let $\alpha = v_0, e_1, \ldots, e_k, v_k$ be a partial soliton walk with respect to state M in a generalized soliton graph G. Then there exists an external M-alternating trail β starting from v_0 such that β terminates in e_k and $E(\beta) \subseteq E(\alpha)$. Furthermore, if $n_\alpha(k)$ is odd, then the other endpoint of β is v_k.*

Proof. First suppose that $e_k \notin M$. Extend G by new external edges $e = (v_{k-1}, v)$ and $e' = (v_k, v')$, such that $v, v' \notin V(G)$. Furthermore let e_m denote the last edge of α for which $e_m = e_k$ and $n_\alpha(m)$ is odd. Observe that either $\alpha[v_0, v_{m-1}] + e$ or $\alpha[v_0, v_{m-1}] + e'$ is a soliton walk – denoted by α' – in $G + e + e'$ depending on whether $v_{m-1} = v_{k-1}$ or $v_{m-1} = v_k$. Thus, based on Theorem 2.3.1, there exists an M-alternating network Γ such that switching on Γ and making α' results in the same state M' of $G + e + e'$. Clearly, Γ will contain an M-alternating crossing β' between the endpoints of α'. Observe that $E(\beta') \subseteq E(\alpha')$, as in M' the status of each edge of β' is exchanged with respect to M. Then replacing e (respectively, e') in β' by e_k, we obtain the required M-alternating trail β.

Now consider the case when $e_k \in M$. Then $e_{k-1} \notin M$, thus by the preceding paragraph, there exists an external M-alternating trail β terminating in e_{k-1} and satisfying the requirements of the proposition. If v_{k-1} is the other endpoint of β, then we obtain a suitable alternating trail by $\beta + e_k$. Observe that, if $n_\alpha(k)$ is odd, as the parities of $n_\alpha(k)$ and $n_\alpha(k-1)$ must be the same, then the above situation holds. Finally, in the other case – when v_{k-2} is the endpoint – β crosses e_k, thus it provides the requested trail. $\diamondsuit$

By the above result it is well-founded to call an edge *viable in state M* of a generalized soliton graph G if it is traversed by an external M-alternating trail of G. However, it was proved in [9] that an edge e is viable in some state of G if and only if e is viable in all states of G. Combining this result with Proposition 2.4.4, we obtain that a non-viable edge of G is necessarily impervious in all states of G. Summarizing the above facts, any edge of a generalized soliton graph G is classified as either *viable*, i.e. traversed by an external alternating trail in all states

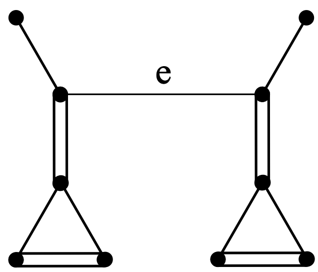

Figure 2.13: An impervious edge e.

of G, or *impervious*, i.e. not traversed by any external alternating trail in any state of G. See Fig. 2.13 for a graph containing an impervious edge e.

The *viable part* of a generalized soliton graph G – denoted by G^+ – is the graph determined by the viable edges of G. Furthermore, we will say that G is *viable* if $G = G^+$. In Figure 2.13, G^+ is determined by the edges different from e.

It is easy to see that any maximal alternating trail in a generalized soliton graph G is enterily contained in G^+, which implies that G^+ is also a generalized soliton graph such that $Ext(G^+) = Ext(G)$ and $M_{(G^+)} \in S(G^+)$ for any state M of G. Conversely, for any $M' \in S(G^+)$ and $M \in S(G)$, $M' \cup (M \setminus M_{(G^+)})$ is clearly a state of G. Summarizing the above facts, we have the following observation.

Proposition 2.4.5 *Let G be a generalized soliton graph and let $S^+ = \{M_{(G^+)} \mid M \in S(G)\}$. Then G^+ is also a generalized soliton graph with $Ext(G) = Ext(G^+)$ and $S(G^+) = S^+$.*

It is clear that any soliton automaton asociated with graph G and state M is strongly isomorphic to the automaton associated with G^+ and $M_{(G^+)}$. Therefore it is justified to use G^+ instead of G in the definition of soliton automata. However, we need one more result so that our approach becomes matching- independent.

Proposition 2.4.6 *Let G be a viable generalized soliton graph and M, M' be distinct states of G. Then there exists a sequence of states $M_1, \dots, M_n$ of G for some $n \geq 2$ such that $M_1 = M$, $M_n = M'$ and for each $1 \leq i \leq n-1$, $M_{i+1} = S(M_i, \alpha_i)$, where α_i is an appropriate soliton walk with respect to M_i.*

Proof. Consider the M-alternating network $\Gamma = \{\beta_1, \dots, \beta_{n-1}\}$ ($n > 1$) by which $M' = S(M, \Gamma)$, let $M_1 = M$, and for the rest of the proof, let i denote an arbitrary index with $i \in [n-1]$. Define M_{i+1} by $M_{i+1} = S(M, \{\beta_1, \dots, \beta_i\})$ and observe that β_i is also an M_i-alternating unit with $M_{i+1} = S(M_i, \beta_i)$. Therefore, if β_i is a crossing, then β_i is a suitable choice for α_i. Otherwise, as G is viable, there exists an external M_i-alternating path β' between some external vertex v and a vertex of β_i. Then an M_i-alternating v-racket γ can be constructed such that $\gamma_c = \beta_i$ and γ_h is the prefix of β' from v to the first vertex of β_i. Now it is easy to see that $\gamma_h + \gamma_c + \gamma_h^{-1}$ provides an appropriate soliton walk as α_i. The index i was arbitrary, thus the proof is complete. $\diamondsuit$

Now we are ready to give the matching theoretic formalization of soliton automata. As our concept is based on generalized soliton graphs, we will use the term "generalized soliton automata".

For the definition we need the following notation:
For any state M of generalized soliton graph G and $v_1, v_2 \in Ext(G)$, let $\mathcal{S}_G(M, v_1, v_2) = \{S(M, \alpha) \mid \alpha \text{ is a soliton walk with respect to } M\text{, which starts at } v_1 \text{ and ends at } v_2\}$

Definition 2.4.7 A *generalized soliton automaton* associated with underlying graph G is a non-deterministic finite automaton

$$\mathcal{A}(G) = ((S(G^+), (X \times X), \delta)$$

subject to the following conditions:

(a) G is a generalized soliton graph

(b) $S(G^+)$, the set of states of $\mathcal{A}(G)$, is the set of states of G^+

(c) $(X \times X)$ is the input alphabet, where $X = Ext(G)$

(d) $\delta : S(G^+) \times (X \times X) \to 2^{S(G^+)}$ is the transition function, such that

$$\delta(M, (v_1, v_2)) = \begin{cases} \mathcal{S}_{G^+}(M, v_1, v_2), & \text{if } \mathcal{S}_{G^+}(M, v_1, v_2) \neq \emptyset \\ \{M\}, & \text{otherwise} \end{cases}$$

for any $M \in S(G^+)$ and $v_1, v_2 \in X$.

By Proposition 2.4.6, any state of a viable graph can be reached from any other state by a sequence of state transitions. Therefore it is easy to see that if we consider generalized soliton graphs as underlying objects in Section 2.2, then Definitions 2.2.4 and 2.4.7 are equivalent up to strong isomorphism. The concepts of determinism and strongly determinism can be also applied for the generalized case in a natural way.

In our closing result, in Theorem 2.4.10, we will prove that the loops and the vertices with degree greater than 3 can be elaborated from any generalized soliton graph, which shows that the concepts of this section and that of Section 2.2 are indeed equivalent. For the above goal, we will apply a shrinking operation on graphs, which was studied first time in [10].

Definition 2.4.8 A *redex* r in graph G consists of two adjacent edges $e = (u, z)$ and $f = (z, v)$ such that $u \neq v$ are both internal and $deg(z) = 2$. The vertex z is called the *center* of r, while u and v (e and f) are the two *focal vertices* (respectively, *focal edges*) of r.

Let r be a redex in G. *Contracting* r in G means creating a new graph G_r from G by deleting the center of r and merging the two focal vertices of r into one vertex s. The vertex s is called the *sink* of r in G_r. Figure 2.14 shows a simple example for a redex, where the notation for the vertices and edges are the same that were used in the above definitions.

Now suppose that G is a generalized soliton graph. For a state M of G, let M_r denote the restriction of M to edges in G_r. Clearly, M_r is a state of G_r. Conversely, if G_r is a generalized soliton graph, then notice that the state M of G can be reconstructed from any $M_r \in S(G_r)$ in a unique way. In other words, G is a generalized soliton graph if and only if G_r is such with the connection $M \mapsto M_r$ being a one-to-one correspondence between the states of G and those of G_r. Graph G and state M will be referred to as the *unfolding* of G_r and M_r, respectively.

For any walk α in G, let $trace_r(\alpha)$ denote the restriction of α to edges in G_r. Obviously, $trace_r(\alpha)$ is a walk in G_r. It is also easy to see that if α is a soliton walk in G with respect to M, then $trace_r(\alpha)$ is a soliton walk in G_r with respect to M_r. Moreover, the soliton walk α can uniquely be recovered from $trace_r(\alpha)$ by unfolding. (Remember that the orientation of loops must be respected in soliton walks.) Consequently, the connection $\alpha \mapsto trace_r(\alpha)$ is also a one-to-one correspondence between soliton walks in G and soliton walks in G_r. Furthermore, $M' = S(M, \alpha)$ holds in G iff $(M')_r = S(M_r, trace_r(\alpha))$ holds in G_r.

The above observations and their immediate consequences are summarized in the following proposition .

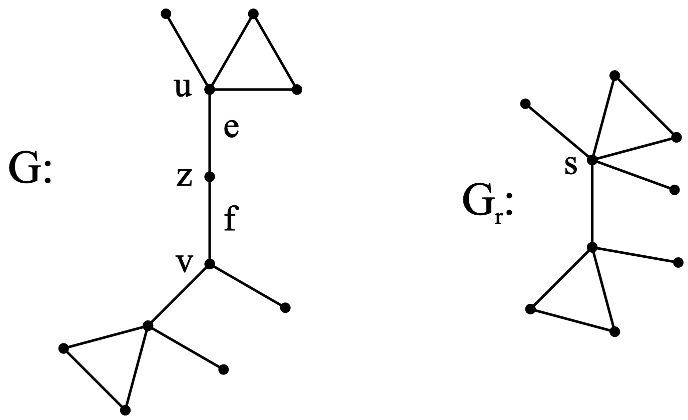

Figure 2.14: Contracting a redex in graph G.

Proposition 2.4.9 *Let r be a redex in graph G. Then G is a generalized soliton graph if and only if G_r is such. Moreover, in this case, $\mathcal{A}(G)$ is strongly isomorphic with $\mathcal{A}(G_r)$, and $\mathcal{A}(G)$ is strongly deterministic if and only if $\mathcal{A}(G_r)$ is such.*

Now we are ready to prove our closing theorem.

Theorem 2.4.10 *Let G be a generalized soliton graph. Then there exists a generalized soliton graph G_1 with the following conditions:*

(a) $\mathcal{A}(G)$ is strongly isomorphic with $\mathcal{A}(G_1)$;

(b) G is strongly deterministic if and only if G_1 is such;

(c) every connected component of G_1 forms an open graph;

(d) G_1 is a loop-free simple graph;

(e) for every $v \in V(G_1)$, $d(v) \leq 3$.

Proof. By Definition 2.4.7, G and G^+ induce the same soliton automaton, thus G^+ satisfies conditions $(a)-(c)$. Now let $e = (v, w)$ be an arbitrary edge of G^+ such that either e is a loop or there exists an edge e' parallel to e. Then subdivide e by new internal vertices v_1 and v_2, i.e. replace e with a trail $v, f_1, v_1, f_2, v_2, f_3, w$, and let G' denote the resulting graph. Observe that f_1, f_2 constitute a redex r in G' with center v_1 and focal vertices v, v_2. Clearly $G'_r = G^+$, hence repeating the above argument for all loops and parallel edges, with applying Proposition 2.4.9, we obtain a graph G'' satisfying conditions $(a)-(d)$.

If (e) also holds for G'', then we are ready. Otherwise, let $v \in Int(G'')$ be a vertex with $d(v) > 3$ and let $v_1, \ldots, v_k$ $(k = d(v))$ be the adjacent vertices of v. Now construct the graph G''' from G'' in the following way: let $V(G''') = V(G'') \setminus \{v\} \cup \{w, w_1, w_2\}$, where $w, w_1, w_2 \notin V(G'')$, and let $E(G''') = E(G'') \setminus \{(v, v_j) \mid j \in [k]\} \cup \{(w, v_1), (w, v_2), (w, w_1), (w_1, w_2)\} \cup \{(w_2, v_i) \mid 3 \leq i \leq k\}$. It is clear that for the redex r having center w_1, $G'''_r = G''$ holds, and the sum of the degrees of vertices v' with $d(v') > 3$ is smaller in G''' than in G''. Therefore, applying the above method with Proposition 2.4.9 in an iterative way, the requested graph G_1 is obtained. $\diamondsuit$

By the above theorem it is meaningful to use the phrases "soliton graphs" and "soliton automata" instead of "generalized soliton graphs" and "generalized soliton automata", respectively, in the rest of the thesis.

Chapter 3

Tutte type characterizations of soliton graphs

3.1 Introduction

In this chapter we will characterize soliton graphs by Tutte type theorems, named after the most important result of non-bipartite matching theory ([97]). The power of Tutte's theorem comes from the fact that it sets forth conditions which are both necessary and sufficient for the existence of a perfect matching in a general graph. This powerful result implies almost all the results on matchings known previously - like Frobenius' Theorem ([47]) and P.Hall's Theorem ([65]).

Motivated by the Berge Formula ([24]), in graphs with perfect matchings, the sets X for which equality holds between $|X|$ and the number of odd components of $G - X$ are called barriers ([85]). It is natural to ask if barriers are meaningful for graphs with perfect internal matchings, and if so, exactly how they work. This question is answered by our first Tutte type theorem in Section 3.3. We propose the concept "splitter" to take over the role of barriers in graphs with perfect internal matchings, and we will show that for any maximal splitter X, the difference between $|X|$ and the number of odd components of $G - X$ containing internal vertices only is at most 1. Furthermore, it is also proved that a maximal splitter can match the power of barriers only if it does not contain accessible vertices.

In [8], the exact counterpart of Tutte's theorem on graphs with perfect matchings has been elaborated for graphs having a perfect internal matching. This theorem establishes a necessary and sufficient condition for a graph G to have a perfect internal matching, requiring that, for any set X of internal vertices, the cardinality of X should not be smaller than the number of connected components of $G-X$ consisting of an odd number of internal vertices. However, the authors of [8] did not study the question of strengthening the above result. Our first Tutte-type theorem partly solves the problem by showing that for maximal splitters as extreme sets, the difference in the above inequality is at most 1. Therefore, in order for the problem to be well-characterized, we only need to describe the graphs for which the inequality is sharp. Our second Tutte type theorem gives the answer by showing that the graphs containing an inaccessible vertex are exactly the ones have the above property.

This chapter is based on the results of [19] and it is organized as follows. In Section 3.2 we generalize the concept of factor-critical graphs, originally defined by Gallai ([52]), in order to cover graphs with perfect internal matchings, and introduce the definition of splitters. Sections 3.3 and 3.4 contain the two Tutte type theorems characterizing soliton graphs in terms of maximal splitters and factor-critical graphs, both of which theorems are significantly stronger than the result in [8].

3.2 Factor-critical graphs and splitters

In this section we generalize the definition of factor-critical graphs given in [6], and introduce the concept “splitter”, which will serve as a basis for our observations in Sections 3.3 and 3.4.

Definition 3.2.1 A connected graph G is *factor-critical* if for every internal vertex v, G has a matching M_v covering every internal vertex but v.

The matching M_v in Definition 3.2.1 above is called a *near-perfect internal matching*, as it covers all internal vertices of G but one. Notice that, according to this definition, a connected graph G is factor-critical if and only if $G - v$ has a perfect internal matching for every internal vertex v.

In Proposition 3.2.5 we give a characterization of factor-critical open graphs with the help of alternating paths. For this we introduce the following concept.

Definition 3.2.2 Let M be a state of soliton graph G and $w \in Ext(G)$. An internal vertex v of G is called *accessible* from w in M (or simply v is M-accessible from w), if there exists a positive external M-alternating path connecting w an v. Furthermore, an internal trail is said to be *M-accessible* from w if some of its vertices is accessible from v in M.

Generally it is not true that if a vertex is accessible from an external vertex w is some state, then it is accessible from w in any state. Nevertheless, as the following claim shows, the accessibility without specifying the external vertex is matching-invariant. For this, we say that ”vertex v is accessible in state M”, by which we mean that v is M-accessible from some external vertex.

Claim 3.2.3 *An internal vertex v is accessible in state M of soliton graph G if and only if v is accessible in all states of G.*

Proof. Let us augment G by a new external edge at v, that is, by an edge $e = (v, v')$, where $v' \notin V(G)$. If $G + e$ denotes the augmented graph, then $G + e$ still has a perfect internal matching, moreover, any perfect internal matching of G is also a perfect internal matching of $G + e$. We shall therefore identify each state of G by its corresponding state of $G + e$. By assumption, there exists an M-alternating crossing α in $G+e$ passing through the edge e. Now let M' be an arbitrary state of G and consider the state $S(M, \alpha)$. It is clear that the mediator alternating network Γ between $S(M, \alpha)$ and M' contains a unique crossing β going through e. Hence stripping β from the edge e results in the desired positive external M'-alternating path in G leading to vertex v. ♢

By virtue of Claim 3.2.3 we can say that an internal vertex v is *accessible* in G without specifying the state M and external vertex v relative to which this concept was originally defined. Moreover, as a consequence of the above result, by Corollary 3.2.4, we obtain an equivalent definition for impervious edges.

Corollary 3.2.4 *An internal edge e is impervious iff neither of its endpoints is accessible.*

Proof. Suppose first that $\alpha = v_0, e_1, v_1, \ldots, e_m, v_m$ $(m > 1)$ is an external alternating trail with $e_m = e$. If e_m is positive in α, then we are ready. Otherwise $\alpha[v_0, v_{m-1}]$ provides a suitable positive alternating path. Conversely, let $\beta = w_0, f_1, w_1, \ldots, f_n, w_n$ $(n > 0)$ be a positive external alternating path such that w_n is an endpoint of e. Then either β itself – in the case of $f_n = e$ – or β together with e will constitute the requested alternating trail. ♢

Proposition 3.2.5 *A connected open graph G is factor-critical if and only if G has a perfect internal matching and every internal vertex is accessible in G.*

Proof. *'Only if':* Let w be any internal vertex incident with some external edge e of G, and consider the matching M_w covering all of $Int(G)$ except w. Then $M_w \cup \{e\}$ is a perfect internal matching of G. Thus, G does have a perfect internal matching.

Now let v be an arbitrary internal vertex, and augment G by a new external edge e incident with v. In the resulting graph $G + e$, $M = M_v \cup \{e\}$ determines a perfect internal matching. On the other hand, every perfect internal matching of G is itself a perfect internal matching for $G + e$ by which e is left uncovered. Thus, by Proposition 2.3.2, there exists an M-alternating crossing β in $G + e$ through e. Switching on β then determines a state M' for G in which v is accessible via β.

'If': For any state M and positive external M-alternating path β leading to v, switching on β determines a suitable near-perfect internal matching of G. ♢

The following lemma will be used in Section 3.3

Lemma 3.2.6 *Let G be a factor-critical open graph with $|Ext(G)| \geq 2$. In all states M of G, every external edge is traversed by a suitable M-alternating crossing.*

Proof. Choose state M and external edge e arbitrarily. If $E(G) = \{e\}$ then we are through. Otherwise let u and v denote the external and internal endpoints of e, respectively. If $e \notin M$, then e, joined with a positive M-alternating path leading to v, forms a crossing in G. Assume therefore that $e \in M$, and let $f = (v, z)$ be any edge adjacent to (but different from) e such that $v \neq z$. Such an edge must exist, as $Ext(G) \geq 2$. Then either z is external, so that ef is a crossing, or there exists a positive external M-alternating path β leading to z. If β starts out from an external vertex different from u, then βef becomes a crossing in G. If β starts out from u, then $\gamma = \beta f$ is an M-alternating loop with a "handle" e.

Let G' be a maximal subgraph of G with the property P that G' is factor critical, $Ext(G') = \{u\}$, and $M_{(G')}$ is a perfect internal matching of G'. Notice that γ has property P, so that a suitable G' exists. Moreover, $G' \neq G$, hence there exists a vertex $x \in V(G) \setminus V(G')$ that is adjacent to some vertex y in G'. Consider a positive external M-alternating path α in G leading to x if x is internal, or take α to be the empty path from x if x is external. We claim that α and G' have no vertices in common. Indeed, if this was not the case, then α would have a suffix α' starting out from an internal vertex of G' and not returning to G' any more before reaching x. Augmenting G' with the ear consisting of α' and the edge (x, y) would then result in a subgraph of G satisfying property P, which contradicts the fact that G' is maximal. We conclude that α, joined with the edge (x, y) and a suitable positive external M-alternating path from u to y within G', forms a crossing in G. ♢

Let u and v be two internal vertices of a graph G having a perfect internal matching. We say that u and v *attract* (*repel*) each other if an extra edge $e = (u, v)$ becomes allowed (respectively, forbidden) in the graph $G + e$.

The above relationship is characterized in Lemma 3.2.7. This characterization uses the concept of *positive alternating fork* by which we mean a pair of vertex-disjoint positive external alternating paths leading to two distinct internal vertices. Although it is somewhat confusing, in the above case we say that these two vertices are *connected* by the fork.

Lemma 3.2.7 *Two internal vertices u and v of a graph G with a perfect internal matching attract each other if and only if u and v can be connected by a positive alternating path or fork in every state of G.*

Proof. Consider the extra edge $e = (v_1, v_2)$ in the graph $G + e$. Since every state of G is also a state of $G + e$, the edge e cannot be mandatory. Therefore e is not forbidden if and only if there exists an M_e-alternating unit passing through e in any state M_e of $G + e$. The above fact is clearly equivalent to saying that e is not forbidden in $G + e$ if and only if there exists an M-alternating unit passing through e in any state M of G. Then returning to graph G, i.e. omitting edge e, the claim becomes obvious. $\diamondsuit$

Now we introduce the concept that is fundamental in our present analysis of open graphs with perfect internal matchings.

Definition 3.2.8 A nonempty set $X \subseteq Int(G)$ of a soliton graph G is a *splitter* if every two vertices of X repel each other in G. The splitter X is *inaccessible* if all of its vertices are such.

For every nonempty set $X \subseteq Int(G)$, let G_X be the graph obtained from G by connecting with an edge every vertex in X with all internal vertices of G, provided that this edge does not already exist in G, and by attaching an external edge to each vertex in X, again only if such an edge is not present in G. If X is a splitter and $G = G_X$, then we say that G is *X-complete.*

Lemma 3.2.9 *For every $X \subseteq Int(G)$, X is a splitter in G if and only if X is a splitter in G_X.*

Proof. Let G_e be the graph obtained from G by adding just one edge e towards constructing G_X. Clearly, it is sufficient to prove that if X is a splitter in G, then it is one in G_e as well. Assume, to the contrary, that X is a splitter in G, yet, two vertices $x, y \in X$ attract each other in G_e. Let M be any state of G_e that is also a state of G, i.e., one by which the edge e is negative. By Lemma 3.2.7, x and y can be connected by a positive M-alternating path or fork β. Leaving out e from β splits β into several subpaths. Since one endpoint of e is in X, it is inevitable that one of these subpaths becomes a positive M-alternating path connecting two vertices in X, or two of them constitute a positive M-alternating fork connecting two such vertices. Either way, this contradicts X being a splitter in G. $\diamondsuit$

Corollary 3.2.10 *For every $X \subseteq Int(G)$, X is a maximal splitter in G iff X is a maximal splitter in G_X.*

Proof. If X is a maximal splitter in G, then X is a splitter in G_X by Lemma 3.2.9. It must also be maximal, because G has fewer edges than G_X, and therefore any splitter $Y \supset X$ would also be a splitter of G. Conversely, if X is a maximal splitter in G_X, then, again by Lemma 3.2.9, X is a splitter in G. If there was a splitter $Y \supset X$ in G, then Y would be a splitter in G_Y, too. However, G_X has fewer edges than G_Y, so that Y would be a splitter in G_X as well. We conclude that X is maximal in G. $\diamondsuit$

3.3 The first Tutte type theorem on maximal splitters

In this section we present our first Tutte type theorem characterizing maximal splitters in soliton graphs. Since trivial connected components play no essential role in this characterization, we shall assume that our graphs do not have such components. This assumption will be held until Theorem 3.4.5, where it is no longer needed and therefore dropped. Moreover, for the formalization of our Tutte type theorems we need the following concepts.

Definition 3.3.1 For any graph G and $X \subseteq Int(G)$, consider the connected components of the subgraph $G - X$. Component K is called *external* if it contains external vertices of G, and *internal* if all vertices of K are internal. Component K is *degenarate external* if it consists of a single external vertex, and K is *odd internal* if it is internal containing an odd number of vertices. The number of internal components (odd internal components) of $G - X$ will be denoted by $c_{\text{in}}(G, X)$ (respectively, $c^{\text{o}}_{\text{in}}(G, X)$).

The following simple technical observation will be used several times in the sequel.

Lemma 3.3.2 *Let X be a splitter in soliton graph G, K be an arbitrary connected component of $G - X$, and β be an alternating path with respect to some state M of G connecting a vertex v in K with a vertex $x \in X$ in such a way that β is positive at its x end. Then, starting from v, β leaves K on an M-positive edge.*

Proof. By way of contradiction, assuming that β leaves K on a negative edge pointing to some vertex $y \in X$ implies that the continuation of β from y to x is positive. This, however, contradicts the fact that X is a splitter. $\diamondsuit$

Theorem 3.3.3 *For a non-empty set X of internal vertices of a soliton graph G, the following two statements are equivalent.*
(i) The set X is a maximal splitter.
(ii) Each non-degenerate component of $G - X$ is factor-critical such that
(iia) $|X| = c_{\text{in}}(G, X) + 1$, or
(iib) $|X| = c_{\text{in}}(G, X)$ with every external component of $G - X$ being degenerate.
Furthermore, condition (iib) holds in (ii) above if and only if X is inaccessible.

Proof. *(i)⇒(ii)* Without loss of generality, we can assume that G is X-complete. Indeed, the difference between the composition of $G - X$ and that of $G_X - X$ is restricted to a number of degenerate external components, which do not affect the validity of *(ii)*. As to the set X, we know by Corollary 3.2.10 that it is a maximal splitter in G if and only if it is such in G_X.

Let E_X denote the set of edges of G connecting a vertex in X with one not in X. For a non-degenerate component K of $G - X$ and state M of G, denote by $E_X(K, M)$ the subset of $E_X \cap M$ consisting of edges incident with K. Furthermore, let $K_M = K + E_X(K, M)$ with the edges $E_X(K, M)$ classified as external in K_M.

We first show that for all components K and any state M of G, K_M is an open factor-critical graph. To this end, by Proposition 3.2.5, it is enough to prove that every internal vertex of K is accessible from within K_M. For, let M and $v \in Int(K)$ be arbitrary. Since X is maximal, v must attract some vertex $x \in X$ in G. Hence, by Lemma 3.2.7, there exists a positive M-alternating path or fork β connecting v with x in G. If β is an alternating fork such that its branch leading to v is entirely contained in K, then we are ready. Otherwise Lemma 3.3.2 implies that, starting from v, the path β (or the branch of β containing v) leaves K on a positive edge. Therefore in the latter case we can also conclude that v is accessible in K_M.

Now we show that the components K are themselves factor-critical. To this end, we first prove by way of contradiction that for all states M and components K, $|E_X(K, M)| \leq 1$.

Let K be internal. By Lemma 3.2.6, $|E_X(K, M)| \geq 2$ implies the existence of a crossing in K_M connecting two external vertices with respect to all states of K_M. Every crossing in K_M with respect to state $M_{(K_M)}$, however, determines a positive M-alternating path in G connecting two vertices of X, which contradicts X being a splitter. Thus, $|E_X(K, M)| \leq 1$. In fact, $|E_X(K, M)| = 1$, for all vertices of K are accessible in K_M.

Now let K be non-degenerate external, and assume that $E_X(K, M) \neq \emptyset$. Again by Lemma 3.2.6, there exists a crossing α in K_M with respect to state $M_{(K_M)}$ involving at least one external vertex not in $Ext(K)$. Considering α as an M-alternating path in G, extend α to an

M-alternating crossing α_1 in G by adding one external edge from E_X at each end, if necessary. Remember that G is X-complete, so that these external edges do exist in G. Making the crossing α_1 in G then creates a state M_1 of G for which $E_X(K, M_1) \subset E_X(K, M)$. Clearly, this procedure can be repeated until we reach $E_X(K, M_k) = \emptyset$ for some state M_k of G. On the other hand, each iteration step affects one or two external edges in E_X, making them part of the newly created state M_i, $1 \leq i \leq k$. Since there cannot be a positive M_i-alternating fork connecting two vertices in X, it is inevitable that $k = 1$ and $|E_X(K, M)| = 1$. Thus, in general, $|E_X(K, M)| \leq 1$ holds in the case of K being external, too. The very same argument shows that there exists at most one non-degenerate external component K for which $|E_X(K, M)| = 1$. Putting this in a yet stronger form, there exists at most one external component K (degenerate or not) connected to a vertex in X by a positive edge.

It is now easy to see that all non-degenerate components K are indeed factor-critical. First let K be external. It can be achieved by the procedure described in the previous paragraph that $E_X(K, M_1) = \emptyset$ for an appropriate state M_1. This indicates that K has a perfect internal matching, and, as we have seen earlier, every internal vertex of K is accessible in K. We conclude by Proposition 3.2.5 that K is factor-critical.

If K is internal, then for every vertex $v \in Int(K)$ there exists a state M of G such that $(v, x) \in M$ for some $x \in X$. Actually, this holds true for all non-degenerate components K (internal or external), due to the fact that v attracts some vertex $x \in X$ in G. (Remember that G is X-complete.) Considering that $|E_X(K, M)| = 1$, it follows that K does indeed have a matching covering all internal vertices but v. Thus, K is factor-critical.

Now we turn to checking conditions *(iia)* and *(iib)*. By the term "vertex $x \in X$ is *taken* by component K" – or, equivalently, "K takes x" in state M – we mean that x is connected to some vertex in K by an M-positive edge. Clearly, every $x \in X$ is taken by exactly one component K in all states of G. The inequality $c_{\text{in}}(G, X) \leq |X| \leq c_{\text{in}}(G, X) + 1$ follows from the fact that, in all states of G, every internal component of $G - X$ takes exactly one vertex from X, while at most one vertex in X is taken by an external component of $G - X$. Finally, as we have seen above, every non-degenerate component K does take a vertex $x \in X$ in some state of G. (Remember that G does not have trivial components, so that every non-degenerate component of $G - X$ contains an internal vertex.) A non-degenerate external component cannot therefore be present in $G - X$ if $|X| = c_{\text{in}}(G, X)$.

(ii)⇒(i) In every state M of G, each internal factor-critical component of $G - X$ must take a vertex from X, leaving at most one vertex in X to be taken by an external component. This rules out the possibility of two distinct vertices of X being connected by a positive edge. The set X is therefore a splitter.

In order to see that X is maximal, we need to show that every internal vertex v of $G - X$ attracts some vertex in X. This will be attested to by the state M^v constructed below. By virtue of Corollary 3.2.10, we can again assume that G is X-complete. Assemble the state M^v through the following steps:

1. Choose a near perfect matching for each internal component of $G - X$, which misses v if v is in that component.

2. Choose a perfect internal matching for each non-degenerate external component K of $G - X$, unless v is in K. If v happens to be in K, then choose a matching for K that covers all internal vertices but v.

3. Couple up the vertices in X with the internal vertices left uncovered in steps 1 and 2 in an arbitrary way.

If all vertices in X are inaccessible, then, by Lemma 3.2.6, no vertex from X can be taken by an external component of $G - X$ in any state of G. This immediately implies that $|X| = c_{\text{in}}(G, X)$. Conversely, if $|X| = c_{\text{in}}(G, X)$, then all vertices of X are taken by the internal components of $G-X$ in every state of G, giving no chance for these vertices to be accessed by a positive external alternating path. (See Lemma 3.3.2.) The proof of Theorem 3.3.3 is now complete. $\diamondsuit$

3.4 The second Tutte type theorem on maximal inaccessible splitters.

In [8] the following counterpart of Tutte's well-known theorem on graphs with perfect matchings was proved for soliton graphs.

Theorem 3.4.1 *An open graph G is a soliton graph if and only if $c^{o}_{\text{in}}(G,X) \leq |X|$, for all $X \subseteq Int(G)$.*

In this section we will strengthen the above result. For this goal we need two lemmas on maximal inaccessible splitters.

Lemma 3.4.2 *Let X be a maximal inaccessible splitter such that all external connected components of $G-X$ are degenerate. Then X is a maximal splitter.*

Proof. Let X be as prescribed by the conditions of the lemma. We must show that every vertex $v \in Int(G-X)$ attracts some vertex in X. Since X is maximal inaccessible, we can assume that v is accessible. Thus, in any state M of G, there exists a positive M-alternating path β leading to v from some external vertex u. Given the fact that X is inaccessible and $\{u\}$ is a degenerate external component of $G-X$, β must start out from u on a negative edge leading to a vertex $x \in X$. Depriving β from this edge results in a positive alternating path connecting v with x, demonstrating that v attracts x. $\diamondsuit$

Lemma 3.4.3 *Let X be a maximal inaccessible splitter, and K be a non-degenerate external connected component of $G-X$. Then all internal vertices of K are accessible from within K in any state of G.*

Proof. Let M be an arbitrary state of G, and suppose by contradiction that K contains internal vertices that are not accessible from within K. The component K is non-degenerate (and non-trivial), implying that internal vertices accessible from within K do exist. Thus, there exist two adjacent internal vertices u and v in K such that u is accessible from within K but v is not. Then v is either inaccessible altogether, and as such must attract some vertex in X, or v is accessible from outside K by a positive external M-alternating path. We conclude by Lemma 3.4.2 that, in either case, there exists a positive M-alternating path β connecting v with a vertex $x \in X$ inside K.

On the other hand, there exists a positive external M-alternating path γ inside K leading to vertex u. The path γ cannot intersect β, because this would make either v or x accessible from within K through appropriate sections of γ and β. But even if γ and β do not intersect, the positive external M-alternating path $\gamma(u,v)\beta$ still makes x accessible, a contradiction. $\diamondsuit$

Corollary 3.4.4 *If X is a maximal inaccessible splitter, then any external component of $G-X$ can only be connected to X by forbidden edges.*

Proof. Immediate by Lemma 3.4.3. $\diamondsuit$

Now we are ready to prove our second Tutte type theorem on maximal inaccessible splitters.

Theorem 3.4.5 *An open graph G is a soliton graph if and only if $c^{o}_{\text{in}}(G,X) \leq |X|$ for all $X \subseteq Int(G)$. Equality may hold for some non-empty X only if not all connected components of G are factor-critical. In this case, the equation is guaranteed by any maximal inaccessible splitter X.*

Proof. The first statement of the theorem is just a reiteration of Theorem 3.4.1. For the second statement we can assume, without loss of generality, that G is connected. If G is a trivial external component by itself, then the statement holds true. Otherwise all our previous results are available for use.

If $|X| = c^{o}_{\text{in}}(G,X)$ holds for some nonempty X, then every vertex of X must be taken by an odd internal component of $G - X$ in all states of G. The graph $G - x$ therefore cannot have a perfect internal matching for any $x \in X$, implying that G is not factor-critical.

It remains to prove that for every maximal inaccessible splitter X of a non-factor-critical soliton graph G, $|X| = c^{o}_{\text{in}}(G,X)$. Clearly, if G is not factor-critical, then it does contain inaccessible vertices, implying that X is non-empty. Moreover, by Corollary 3.4.4, we can simply delete the non-degenerate external components of $G - X$ from G, preserving just the forbidden edges that connect these components to X as external edges in the simplified graph G'. These external edges remain forbidden in G'. Indeed, assume to the contrary that G' has a state M' by which one or more external edge of G' is positive. Considering the state $M_{(G')}$ for any state M of G, Proposition 2.3.2 implies that there exists a crossing α in G' with respect to $M_{(G')}$. The crossing α, however, determines a positive M-alternating path in G connecting two vertices of X, which is impossible. It follows from this argument that an extra internal edge e added to G' is forbidden in $G' + e$ if and only if it is such in $G + e$. Consequently, two vertices attract each other in G' if and only if they do so in G. Furthermore, by Lemma 3.4.3 and Corollary 3.4.4, it is clear that a vertex of $Int(G')$ is accessible in G' if and only if it is such in G. Thus, X is a maximal inaccessible splitter in G'.

Now the desired statement is obtained by the direct application of Lemma 3.4.2 and Theorem 3.3.3 for the graph G'. $\diamondsuit$

Our concluding observation provides a characterization of factor-critical open graphs.

Corollary 3.4.6 *A connected open graph G is factor-critical if and only if $c^{o}_{\text{in}}(G,X) \leq |X| - 1$ for all non-empty sets $X \subseteq Int(G)$. In this case, equality holds for any maximal splitter X.*

Proof. Immediate by Theorems 3.3.3 and 3.4.5 $\diamondsuit$

Chapter 4

A structure theory for soliton graphs

4.1 Introduction

Compositions and decompositions of finite automata have been intensively studied since the beginning of the sixties. The goal of this research is to characterize complex systems by products of smaller automata. In order to carry out this task for soliton automata, first we need to work out a decomposition of soliton graphs into smaller components such that the automata associated with these components should operate partly independently, i.e. the relationship among the components can be fully described. To meet the above goal, in this section we develop a structure theory of soliton graphs on the basis of their elementary components. These results will then be in focus in the actual automata decomposition developed in Chapter 5.

Elementary graphs with respect to perfect matchings, graphs remaining connected after removing their forbidden edges, have played a central role in classical graph theory. The term "elementary" was coined by Hetyei in the 1960's (cf. [66]), although the idea itself is much older, it was first employed by König in his pioneer paper on matching theory ([73]). One of the main characteristics of these graphs is that a canonical partition can be defined on their vertex set. Historically, this idea seems to have originated from Kotzig ([74],[75],[76]) and was further developed by Lovász ([83]) who proved that the blocks of this partition are exactly the maximal barriers. The concept of elementary graphs with their canonical partition was extended in [9]. Section 4.2 will be devoted to review basic notions and notation relating to elementary graphs, and it puts forward two simple claims for the further parts of the chapter.

In the light of Theorem 3.4.5 it is clear that maximal barriers are not suitable to determine such an equivalence in soliton graphs, but as was shown in [9], splitters can take over the above role. As the first step in the development of our structure theory, in Section 4.4 we will generalize the canonical partition for all soliton graphs. We will prove that the restriction of splitters on the elementary components –maximal subgraphs spanned by allowed edges only– defines a canonical equivalence. The proof will use the technique of Section 4.3 elaborated for perfect internal matchings to be studied in terms of perfect matchings.

Using the above results, we will distinguish between two kinds of elementary components containing viable edges. An elementary component C will be called one-way or two-way depending on whether one or more canonical classes P of C have the property that there exist an external alternating path which reaches C at a vertex in P. The unique class of one-way components containing internal vertices only is called principal. Elementary components containing external vertices are also considered one-way components. It is proved in Section 4.5

that the viable elementary components can be grouped into pairwise disjoint families based on this relationship. More precisely, we will show the followings.

(i) Each family contains a unique one-way elementary component, called the root of the family.

(ii) Any external alternating trail leading to an elementary component C must reach the root of the family containing C first.

(iii) There exists a partial order $\overset{*}{\mapsto}$ among the families reflecting the order by which external alternating paths reach the families. The maximal elements are the families containing an external vertex.

(iv) An edge incident with a vertex in the viable part of a soliton graph is impervious iff both of its endpoints belong to the principal canonical class of some one-way internal elementary component.

Section 4.6 provides a link between the above results and that of Chapter 3 by characterizing families of soliton graphs in terms of splitters and factor-critical graphs. For this goal we define self-contained graphs as viable soliton graphs containing a unique non-degenerate family. Furthermore, the concept of complete splitters is also introduced by which we mean splitters S consisting of vertices attracting all vertices not in S. The main result of this section is that a graph is self-contained iff it is factor-critical or its inaccessible vertices form a complete splitter.

The characterization given in Section 4.6 is then used in an algorithm to isolate the families of a given soliton graph G. The algorithm, which is a simplified version of the Edmonds algorithm ([40]), is described in Section 4.7, and its time complexity is proportional to the number of edges in G, provided that a perfect internal matching has previously been found for G.

Finally we note that the name "family" is justified [12] by showing that the members of a family can be structured into a family tree according to the immediate predecessor relationship. This relationship is also based on the order of the accessibility of elementary components by external alternating paths.

The results of this chapter appear in [17] (Sections 4.2-4.5) and in [15] (Sections 4.6, 4.7).

4.2 Elementary components in soliton graphs

Elementary graphs and elementary components were originally studied with respect to perfect matchings ([85]). Based on [9] and [17], in this section we generalize the related concepts for graphs with perfect internal matchings and state a few claims that will be used in later sections.

Assume, for the rest of this section, that G is a graph having a perfect internal matching. Graph G is called *elementary* if its allowed edges form a connected subgraph covering all the external vertices, and G is *1-extendable* if all of its edges, except the loops if any, are allowed. A subgraph G' of G is *nice* if it has a perfect internal matching, and every perfect internal matching of G' can be extended to a perfect internal matching of G. In this case, a perfect internal matching of G is *G'-permissible* if it is the extension of an appropriate perfect internal matching of G'. Obviously, not all perfect internal matchings of G must be G'-permissible. Take, for example, a single non-constant internal edge e in G (with a loop around both endpoints) as G'. Clearly, G' is nice, but a perfect internal matching M of G is G'-permissible iff $e \in M$.

In general, the subgraph of G determined by its allowed edges has several connected components, which are called the *elementary components* of G. An elementary component C is *external* if it contains external vertices of G, otherwise C is *internal*. Notice that an elementary component can be as small as a single external vertex of G. Such elementary components are called *degenerate* , and they are the only exception from the general rule that each elementary component is an elementary graph. A *mandatory elementary component* is a single mandatory edge $e \in E(G)$ with a loop around one or both of its endpoints, depending on whether e is external or internal. Moreover, an elementary graph is called *mandatory*if its unique elementary component is such.

Claim 4.2.1 *Every internal vertex of an open elementary graph G is accessible.*

Proof. It was proved in [9] that for every two allowed edges e_1, e_2 of an elementary graph there exists a state M such that both e_1 and e_2 are contained in an appropriate M-alternating unit. Let v be an arbitrary internal vertex of G. Clearly, there exists an edge $e \in M$ incident with v. If e is external, then we are through. Otherwise, since e is allowed, for any external edge e' of G there exists a state M' and a crossing α with respect to M' such that α goes through e and e'. Thus, v is indeed accessible (i.e. in state M'). $\diamondsuit$

Claim 4.2.2 *Let C_1 and C_2 be two different external elementary components of G. There exists no negative alternating path β with respect to any state M connecting C_1 and C_2 in such a way that the two endpoints of β, but no other vertices, lie in C_1 and C_2.*

Proof. Indeed, if there was such a path β connecting vertex $v_1 \in C_1$ with vertex $v_2 \in C_2$, then, since both v_1 and v_2 are internal, these vertices would be accessible in state M within their own elementary components by external M-alternating paths α_1 and α_2. (See 4.2.1 above.) Combining α_1, β, and α_2 would then result in a crossing through both components C_1 and C_2, which contradicts that $C_1 \neq C_2$. $\diamondsuit$

4.3 The closure of open graphs

In order to prove a result on open graphs and perfect internal matchings it is sometimes useful to start reasoning about some related closed graphs with perfect matchings, and then deduce the desired result by reopening these graphs. The closure operation introduced in this section allows a deduction mechanism of this nature. Throughout this section, unless otherwise stated, G will denote an open graph.

Definition 4.3.1 The *closure* of graph G is the closed graph G^* for which:
— $V(G^*) = V(G)$ if $|V(G)|$ is even, and
$V(G^*) = V(G) \cup \{c\}$, $c \notin V(G)$ if $|V(G)|$ is odd;
— $E(G^*) = E(G) \cup \{(v_1, v_2) | v_i \in \mathrm{Ext}(G) \cup \{c\}\}$.

Intuitively, G^* is obtained from G by connecting all of its external vertices with each other in all possible ways. If $|V(G)|$ happens to be odd, then a new vertex c is added to G, and edges are introduced from c to all the external vertices. The edges of G^* belonging to $E(G^*) - E(G)$ will be called *marginal*, and the vertex c will be referred to as the *collector*. Edges incident with the collector vertex will also be called *collector edges*.

Notice that, in the specification of $E(G^*)$, it is not required that $v_1 \neq v_2$. Consequently, in G^*, we are going to have a loop around each external vertex of G. These loops have no specific purpose if G has at least two external vertices, although their introduction as trivial forbidden edges is harmless. If there is only one external vertex in G, however, the loop is essential to make G^* closed.

Proposition 4.3.2 *Graph G has a perfect internal matching iff G^* has a perfect matching.*

Proof. If G^* has a perfect matching M^*, then deleting the marginal edges from G^* and M^* will leave G with a perfect internal matching. Conversely, if G has a perfect internal matching M, then it is always possible to extend M to a perfect matching of G^* by matching up the external vertices of G not covered by M in an arbitrary way, using the collector vertex c if necessary. Obviously, the use of c is necessary if and only if $|V(G)|$ is odd. ♢

Lemma 4.3.3 *Every M-alternating crossing of G can be turned into an M^*-alternating cycle of G^* by any extension of M to a perfect matching M^*. Conversely, for an arbitrary perfect matching M^* of G^*, every M^*-alternating cycle of G^* containing at least one marginal edge opens up to a number of alternating crosses with respect to the restriction of M^* to $E(G)$ when the marginal edges are deleted from G^*.*

Proof. Straightforward, using the same argument as under Proposition 4.3.2 ♢

Corollary 4.3.4 *For every edge $e \in E(G)$, e is allowed in G iff e is allowed in G^*.*

Proof. Indeed, by Lemma 4.3.3,
e is allowed in G
iff there exists a M-alternating unit through e in G for some M
iff there is an M^*-alternating cycle through e in G^* for some M^*
iff e is allowed in G^*. ♢

Corollary 4.3.5 *For any vertices $u, v \in Int(G)$, u and v attract each other in G iff they attract each other in G^*.*

Proof. As clearly $(G + (u, v))^* = G^* + (u, v)$, the argument is immediate by Corollary 4.3.4. ♢

Corollary 4.3.6 *A connected graph G is elementary iff G^* is elementary.*

Proof. If G is elementary, then its allowed edges form a connected subgraph G_e of G covering all the external vertices. By virtue of Corollary 4.3.4, G_e is part of an elementary component in G^*, which must be the only one as the collector vertex alone cannot form an elementary component in the closed graph G^*. Conversely, let G^* be elementary, and assume by way of contradiction that G has more than one elementary components. All these components must be external, because any internal elementary component of G would also be an elementary component of G^* according to Corollary 4.3.4 . Since G is connected, there must be two elementary components in G that are connected by a forbidden edge, which is in contradiction with Claim 4.2.2. ♢

By Corollary 4.3.4, if the closure G^* of a connected graph G is 1-extendable, then so is G. Conversely, if G is 1-extendable, then only the marginal edges of G^* might be forbidden in G^*. Among these, however, the collector edges are ruled out for the following reason. Let v be an arbitrary external vertex of G, and consider a state M of G by which v is left uncovered. Such a state M can always be found, because if a randomly chosen M' does cover v, then switching to state $M = S(M', \alpha)$ for an appropriate crossing α starting from v will do the job. (Crossing α will exist, for G cannot be a a single mandatory external edge if the collector vertex is present.) Now we can extend M to a perfect matching M^* of G^* by first putting in the edge (v, c), then matching up the remaining uncovered external vertices of G in an arbitrary way. This proves the edge (v, c) allowed. Thus, only those marginal edges can be forbidden in G^* that connect the external vertices of G directly. Fig. 4.1 shows a simple example where all these edges are indeed forbidden.

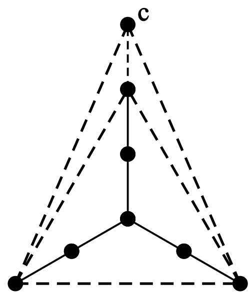

Figure 4.1: Marginal edges that are forbidden in G^*.

If G is not elementary, then several of its external elementary components may be amalgamated in G^*. The internal elementary components of G, however, will remain intact in G^* as every forbidden edge of G is still forbidden in G^*. The mandatory external elementary components of G, too, will remain mandatory in G^*. We claim that the union of all non-mandatory external elementary components of G, together with the collector vertex if that is present, forms one elementary component in G^*, called the *amalgamated* elementary component. Indeed, as we have already seen, every collector edge not adjacent to a mandatory external edge of G is allowed in G^*. Similarly, if e is an edge in G^* connecting two external vertices of G belonging to different non-mandatory elementary components, then it is always possible to find a state M of G by which the two endpoints of e are not covered. Then M can be extended to a perfect matching M^* of G^* by putting in the edge e first, so verifying it to be allowed in G^*.

The observations of the previous paragraph are summarized in Theorem 4.3.7 below, which provides a characterization of the elementary decomposition of G^*.

Theorem 4.3.7 *The set of elementary components of G^* consists of:*

(i) the internal elementary components of G;

(ii) the mandatory external elementary components of G;

(iii) the amalgamated elementary component, which is the union of all non-mandatory external elementary components of G and the collector vertex, if that is present.

4.4 Canonical equivalence

In [9] was proved that the maximal splitters of any elementary graph G defines a *canonical partition* of $Int(G)$, the blocks of which are called *canonical classes*. The equivalence relation corresponding to the above partition is called *canonical equivalence* and denoted by $\sim$. We generalize this relation for non-elementary graphs in the following natural way.

Definition 4.4.1 Let G be a graph having a perfect internal matching. Then for any two internal vertices $u, v \in V(G)$, $u \sim v$ if u and v repel each other and they belong to the same elementary component of G.

One might think that the relation $\sim$, when restricted to a particular elementary component C, results in the equivalence $\sim_C$, which is canonical equivalence on C alone in the usual sense. In general this fails to hold, and we shall see that $\sim |C$ — the restriction of $\sim$ to C — is just a refinement of $\sim_C$. At the moment, however, we do not even know that $\sim$ is an equivalence relation for non-elementary graphs. All we know is that $\sim$ is reflexive and symmetric, and that $u \not\sim_C v$ implies $u \not\sim v$, i.e., $\sim |C \subseteq \sim_C$.

In the light of Lemmas 3.2.7 and 4.3.3, and Corollary 4.3.4it is easy to see that for any two internal vertices u and v belonging to the same elementary component of G, $u \sim v$ holds in G iff $u \sim v$ holds in G^*. Furthermore, if u and v are arbitrary vertices belonging to different non-mandatory external elementary components, then $u \not\sim v$ holds in G^*. Indeed, by Claim 4.2.1, if u (v) is internal, then there exists a positive external alternating path leading to that vertex in its elementary component with respect to any state M of G, which path (paths) will give rise to a positive alternating path connecting u with v in G^* with respect to any extension of M. Finally, by the same argument, $c \not\sim v$ holds for the collector vertex c and any other vertex v in the amalgamated elementary component of G^*. Thus, we have proved the following characterization of the relation $\sim_{G^*}$ in terms of $\sim_G$.

Theorem 4.4.2 *Let u and v be vertices of an elementary component C in G^*.*

(i) If u and v are both internal in G, then, irrespective of the choice of C, $u \sim_{G^*} v$ iff u and v are in the same elementary component of G, too, and $u \sim_G v$.

(ii) If C is a mandatory external elementary component of G, then $u \sim_{G^*} v$ iff $u = v$.

(iii) For C being the amalgamated elementary component, $u \not\sim_{G^*} v$ whenever u and v belong to different external elementary components of G, or exactly one of them is the collector vertex. If u and v are external vertices of the same elementary component in G, then either of $u \sim_{G^*} v$ and $u \not\sim_{G^*} v$ is possible.

(iv) Statements (i)–(iii) remain true if we replace $\sim_G$ and $\sim_{G^*}$ in them by the local relations $\sim_C$ in G and G^*, respectively.

Corollary 4.4.3 *For every elementary component C of G,*

$$\sim_G |C = \sim_{G^*} |C \quad \textit{and} \quad \sim_C = \sim_{C^*} |C,$$

where C^* is the elementary component of G^* containing C.

Proof. Straightforward by Theorem 4.4.2 (i) and (iv). $\diamondsuit$

Let C be a nice elementary subgraph of G, and consider a C-permissible perfect internal matching M in G. An *M-alternating C-loop* (or just C-loop if M is understood) is a negative internal M-alternating path or loop in G having both endpoints, but no other vertices, in C. See Fig. 4.2a. If C is closed, then an *M-alternating C-fork* is a pair of edge-disjoint negative external M-alternating paths such that their internal endpoints, but no other vertices, are in C. See Fig. 4.2b. A C-loop (fork) is said to *connect* its internal endpoints even if this does not in fact happen in the case of forks.

Definition 4.4.4 A *hidden edge* of G is an edge $e = (v_1, v_2)$, not necessarily in $E(G)$, for which v_1 and v_2 are the endpoints of an M-alternating C-loop or C-fork for some elementary component C and state M of G. The word "*shortcut*" will sometimes be used as a synonym for "hidden edge".

The word "shortcut" is often used as a synonym for "hidden edge". Note that, by definition, every forbidden edge in an elementary component C of G is a C-loop, and hence becomes a hidden edge of G. Reversing the argument one can see that hidden edges always become forbidden in their respective elementary components. Indeed, suppose that $v_1 \not\sim_C v_2$ for the two endpoints v_1 and v_2 of an M-alternating C-loop or C-fork α. Then there exists a positive M-alternating path or fork β connecting v_1 with v_2 running entirely in C. (See Lemma 3.2.7.) Notice that a fork α cannot be coupled with a fork β in one case, since an alternating C-fork

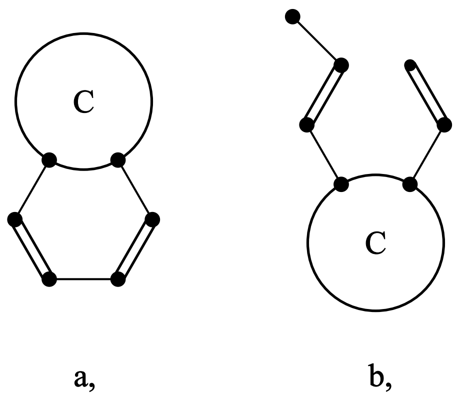

Figure 4.2: A C-loop and a C-fork.

exists only if C is closed. Combining the negative α with the positive β then results in an M-alternating unit in G containing α, which contradicts the fact that C is an elementary component.

Let us now have a closer look at the composition of an alternating C-loop α for some elementary component C. Intuitively, α starts out from an internal vertex of C and, after traversing a forbidden edge of G, enters another elementary component C_1. After making a positive alternating path in C_1 the whole process is iterated, so that by the time α returns to C, a sequence $C_1, \ldots, C_n$ of elementary components will have been visited. Note that the case $n = 0$ is possible, indicating the presence of a single forbidden edge in C as a C-loop. Also notice that there might be repetitions in the sequence $C_1, \ldots, C_n$, as any of these components can be left and reentered subsequently. We say that the components $C_1, \ldots, C_n$ are *covered* by the C-loop α. Component C_i is covered with *multiplicity* m_i if α visits C_i exactly m_i times.

The following proposition shows that the particular matching M, relative to which α is defined, has no bearing on the existence and composition of C-loops covering internal components only.

Proposition 4.4.5 *Let α be a C-loop connecting vertices v_1 and v_2 of an elementary component C with respect to some state M of G, and assume that all components covered by C are internal. Then, for every state M', there exists an M'-alternating C-loop connecting v_1 and v_2 that goes through the same forbidden edges as α and covers the same set of elementary components with the same multiplicity, too.*

Proof. Let $\mathcal{C}_\alpha$ be the set of elementary components covered by α, and consider the subgraph $G[\cup\mathcal{C}_\alpha]$ of G determined by the union of these components. Augment $G[\cup\mathcal{C}_\alpha]$ by the two forbidden edges e_1 and e_2 of α originally incident with v_1 and v_2, and consider them as external edges. Denote the resulting graph having two external vertices by G_α, and let M_α (M'_α) be the restriction of M (respectively, M') to G_α. Clearly, G_α is elementary, since the opening of the loop α — being an M_α-alternating crossing in this graph — connects the components in $\mathcal{C}_\alpha$ to each other. Consider the state $S(M_\alpha, \alpha)$ of G_α. Making the crossing α in this state and then switching to state M'_α determines an alternating network N with respect to state M'_α. The network N will consist of several cycles within the components belonging to $\mathcal{C}_\alpha$ and one crossing α' connecting the two external vertices. Clearly, the crossing α' determines a C-loop in G with respect to state M'. All the forbidden edges of G traversed by α will also be traversed by α', as none of these edges are present in either M or M'. Thus, α' covers exactly the same elementary components as α, not necessarily in the same order, though. Nevertheless, it certainly covers each one with the same multiplicity as α. $\diamondsuit$

Lemma 4.4.6 *The hidden edges of G^* different from the forbidden marginal edges are exactly the hidden edges of G.*

Proof. Let α be a C-loop or C-fork in G for some elementary component C with respect to state M. If C is internal, then obviously α determines a C-loop α^* in G^* with respect to any extension of M to a perfect matching M^*. If C is external, then Claim 4.2.2 implies that α is a loop that will not reach any other external elementary component of G. Therefore α^* becomes an A-loop in G^*, where $A = C$ if C is mandatory, and A is the amalgamated elementary component otherwise. Thus, every hidden edge in G is one in G^*.

Now let α be a C-loop connecting vertices v_1 and v_2 of an elementary component C in G^* with respect to some perfect matching M^*. By Theorem 4.4.2, neither v_1 nor v_2 is the collector. If either v_1 or v_2, say v_1, is external in G, then v_2 is external, too, belonging to the same elementary component of G as v_1. Indeed, by Theorem 4.4.2, there are no forbidden edges in G^* incident with v_1 other than the marginal ones. Let therefore v_1 and v_2 be both internal in G. By Claim 4.2.2, these two vertices are in the same elementary component of G even if $C = A$ is the amalgamated elementary component. Therefore there exists an elementary component C' of G such that either α is a C'-loop or it opens up to a C'-fork with respect to the restriction of M^* to G. Thus, every hidden edge of G^* that is not a forbidden marginal edge is a hidden edge of G. $\diamondsuit$

For every elementary component C of G, let C_h denote the enhancement of C with all the hidden edges belonging to C. Similarly, denote by G_h the graph obtained from G by adding all of its hidden edges.

Corollary 4.4.7

$$(G_h)^* = (G^*)_h.$$

Proof. Straightforward by Lemma 4.4.6. $\diamondsuit$

Corollary 4.4.8 *Let α be a C-loop or C-fork connecting vertices v_1 and v_2 of some elementary component C with respect to state M of G. Then for every state M' there exists an M'-alternating C-loop or C-fork connecting v_1 and v_2 that covers the same forbidden edges and elementary components as α.*

Proof. By Proposition 4.4.5 it is enough to prove the statement in the case when α is either a fork or it is a loop covering an external elementary component D. Claim 4.2.2 then implies that C is internal and D is unique. Let M^* and $(M')^*$ be any extensions of M and M' to perfect matchings in G^*. Following the argument in the first paragraph of the proof of Lemma 4.4.6, α determines an appropriate C-loop α^* in G^* with respect to M^*. Using Proposition 4.4.5 again, there exists a C-loop $(\alpha')^*$ with respect to $(M')^*$ in G^* covering the same forbidden edges and elementary components as α^*. Reopening G^* then determines a C-loop or C-fork α' with respect to M' in G. Since the external component D that might affect the opening of $(\alpha')^*$ into α' is unique, α' will cover the same forbidden edges and elementary components as α. $\diamondsuit$

Our goal is to show that the elementary decomposition of G_h is the same as that of G, and all the hidden edges of G remain forbidden in G_h. Although this fact might seem obvious to the reader already at this point, its formal proof poses a technical challenge, which will be dealt with in Lemma 4.4.9 and Theorem 4.4.11 below.

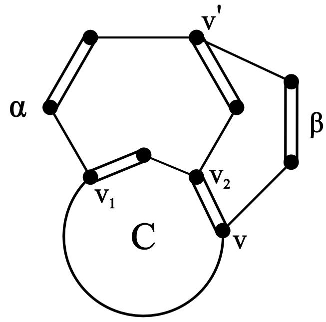

Figure 4.3: The proof of Lemma 4.4.9.

Let C be any elementary subgraph of G, and assume that a negative alternating trail α is such that none of its vertices, except possibly the endpoints, are in C. We shall refer to this situation by saying that α runs *essentially outside* C.

Lemma 4.4.9 *Let C be a nice elementary subgraph of G, and let $v, v_1, v_2 \in V(C)$ be such that $v_1 \sim_C v_2$ but $v \not\sim_C v_i$ for $i = 1, 2$. Moreover, for some C-permissible state M of G, let α be an M-alternating C-loop or C-fork connecting v_1 and v_2, and β be a negative M-alternating path running essentially outside $C + \alpha$, connecting v with a vertex v' lying on α. Then there exists an M-alternating unit in $C + \alpha + \beta$ containing β.*

Proof. *(i)* Assume first that G is closed, so that α is a C-loop. The situation is depicted by Fig. 4.3. The edge $e \in M$ on α incident with v' acts like a valve for β in the sense that it points to either v_1 or v_2. Say the valve points to v_2 as in Fig. nicesubgraph-lemmaproof-figure. Let γ be the M-alternating path that starts out from v on β, then switches to α at v', and ends in v_2. Since $v_2 \not\sim_C v$, there exists a positive M-alternating path connecting v_2 with v inside C. Combining this path with the negative alternating path γ results in the desired M-alternating cycle.

(ii) If G is open, then consider the closure $[C]^*$ of the subgraph $[C]$ ($= G[C]$), and observe that $[C]^*$ is a nice elementary subgraph of G^*. This is obvious if the collector vertex is present in G^*. If it is not, but the collector is needed for $[C]^*$, then any external vertex of G not in C is suitable for this purpose. Such a vertex will always exist, otherwise the collector would not be necessary in $[C]^*$ either. Clearly, $v_1 \sim_{[C]^*} v_2$ and $v \not\sim_{[C]^*} v_i$ for $i = 1, 2$. Moreover, α determines a $[C]^*$-loop α^* in G^* with respect to any $[C]^*$-permissible extension of M to a perfect matching. This is true because at most one of $[C]^* \neq C$ and $\alpha^* \neq \alpha$ can hold, keeping α^* essentially outside $[C]^*$. (Remember that C must be internal for any C-fork.) Now the statement follows easily from *(i)*. $\diamondsuit$

Corollary 4.4.10 *Let α be an M-alternating C-loop or C-fork for some elementary component C of G connecting vertices v_1 and v_2, and let β be an M-alternating path starting out from a vertex v in C, but running essentially outside C. If $v \not\sim_C v_i$ for both $i = 1, 2$, then β must avoid all the elementary components covered by α.*

Proof. Assume, on the contrary, that there exists a negative M-alternating path β satisfying the conditions of the corollary in such a way that the other endpoint u of β lies on an elementary component C' covered by α, but β runs essentially outside $C \cup C'$. By switching to G^* we can assume, without loss of generality, that α is a loop. (See Lemma 4.4.6.) According to Lemma 4.4.9, β and α cannot have a vertex in common. Let u_1 and u_2 be two vertices of C' where

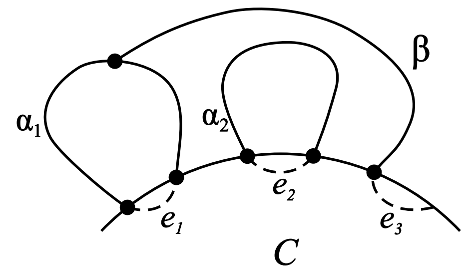

Figure 4.4: Unfolding the loops in Theorem 4.4.11.

α enters and subsequently leaves this component. Clearly, $u_1 \not\sim_{C'} u_2$, so that u and at least one of u_1, u_2 are in different canonical classes by $\sim_{C'}$. The path β can therefore be continued from u inside C' in an M-alternating way to reach u_1 or u_2. In either way this continuation will eventually hit the loop α, which is in contradiction with Lemma 4.4.9. $\diamondsuit$

Theorem 4.4.11 *For an elementary component C of G, let $e_1, \ldots, e_n$ be any number of hidden edges in C. Then, for the elementary graph $C_n = C + e_1 + \ldots + e_n$, each edge e_i remains forbidden in C_n, and $\sim |C \subseteq \sim_{C_n}$.*

Proof. *(i)* Again, assume first that G is closed. The proof is an induction argument on n. For $n = 0$ the statement is trivial. Assume it holds for any choice of hidden edges $e_1, \ldots, e_n$, $n \geq 0$, and let e_{n+1} be a further hidden edge. Let β be an arbitrary positive alternating path or alternating cycle in C_{n+1} with respect to some state M, and try to replace the edges e_i on β by appropriate C-loops one-by-one, until an overlap occurs between two of them in G. Note that such loops always exist by Corollary 4.4.8. We claim that the process of unfolding the hidden edges in β will be successful all the way, that is, all newly introduced C-loops will be pairwise disjoint. On the contrary, let us assume that we encounter an overlap when introducing a C-loop for edge e_i with the one that has been substituted for e_j previously, and this is the first time an overlap occurs. Without loss of generality we can assume that the hidden edges that have already been successfully replaced are $e_1, \ldots, e_{i-1}$, and $j = 1$. See Fig. 4.4.

In the way described above, we will have an instance of the situation captured by Lemma 4.4.9 with C in that lemma being $C^i = C + e_2 + \ldots + e_{i-1}$ now, α being the loop that replaced e_1 with endpoints v_1, v_2, and β being an appropriate subpath of the loop attempted to be substituted for e_i starting out from vertex v. Note, however, that the base graph G in that lemma is now $G+e_2+\ldots+e_{i-1}$, in which we do not know yet if C^i is an elementary component. But it certainly is a nice elementary subgraph. To verify the conditions of the lemma, observe that $v_1 \sim_{C^i} v_2$, since e_1 is still forbidden in $C^i + e_1 = C_{i-1}$ by the induction hypothesis. Moreover, $v_2 \not\sim_{C^i} v$, since there exists a positive alternating path connecting v_2 with v in G using the pairwise disjoint C-loops introduced for $e_2, \ldots, e_{i-1}$, therefore there exists one in $C + e_2 + \ldots + e_{i-1}$ without using them. The application of Lemma 4.4.9 then results in an M-alternating cycle γ in $C^i + \alpha + \beta$ containing β. As β does not overlap with the previously introduced loops for e_k, $2 \leq k \leq i-1$, these loops can be reintroduced in γ to obtain a M-alternating cycle already in G containing β, which is a contradiction.

Having made the above powerful argument, the induction started under *(i)* can now be finished easily. Suppose e_{n+1} becomes allowable in the graph $C + e_1 + \ldots + e_{n+1}$. Then there is an M-alternating cycle γ containing some (in fact all) of the edges e_i, $1 \leq i \leq n+1$. Replacing these edges by appropriate pairwise disjoint C-loops yields an M-alternating cycle in G covering forbidden edges, which is impossible. The proof of $\sim |C \subseteq \sim_{C_{n+1}}$ follows exactly the same argument, and is left to the reader.

(ii) If G is open, then switch to the graph G^*, and apply part *(i)* for this graph and its elementary component C^* containing C. Theorem 4.3.7 and Lemma 4.4.6 ensure that all the required conditions are met. Thus, the edges $e_1, \ldots, e_n$ are forbidden in C_n^*, and $\sim_{G^*} |C^* \subseteq \sim_{C_n^*}$. Coming back to the graph G it follows immediately that $e_1, \ldots, e_n$ are forbidden in C_n. Furthermore,

$$\begin{aligned}
&\sim_G |C \\
&\quad = \sim_{G^*} |C \quad \text{(by Corollary 4.4.3)} \\
&\quad = (\sim_{G^*} |C^*)|C \\
&\quad \subseteq \sim_{C_n^*} |C \\
&\quad = \sim_{C_n} \quad \text{(by Corollary 4.4.3).}
\end{aligned}$$ ♢

Corollary 4.4.12 *For every elementary component C,*

$$\sim |C = \sim_{C_h} .$$

Proof. Notice that $\sim_{C_h} \subseteq \sim |C$, because every positive alternating path or fork β in G connecting two vertices of C can be turned into a path or fork β_h in C_h by making the appropriate shortcuts. This fact is obvious unless C is external and β is a fork. But in this case, too, Claim 4.2.2 implies that β_h remains in the elementary component C_h. On the other hand, $\sim |C \subseteq \sim_{C_h}$ follows from Theorem 4.4.11. ♢

Corollary 4.4.13 *The elementary decomposition of G is the same as that of G_h.*

Proof. It is sufficient to prove that the addition of just one hidden edge e to G does not change the elementary decomposition of G. This is equivalent to saying that e is forbidden in $G + e$. Suppose, by contradiction, that for any hidden edge e connecting vertices v_1 and v_2 in elementary component C there exists an inter-elementary alternating unit γ in $G + e$ with respect to some state M of $G + e$ going through e. Without loss of generality we can assume that $e \notin M$, i.e., M is a state of G, too. The unit γ puts v_1 and v_2 in different canonical classes according to $\sim |C$. But then, by Corollary 4.4.12, v_1 and v_2 cannot be in the same canonical class according to $\sim_{C_h}$ either, which is in contradiction with Theorem 4.4.11. ♢

The key observation made in the proof of Theorem 4.4.11 is now generalized and stated as a separate principle.

Theorem 4.4.14 (*Shortcut Principle*) *For any state M, let γ be an arbitrary M-alternating trail in G_h. Then any number of the shortcuts along γ can be unfolded into appropriate M-alternating loops or forks without the chance of creating any intersections. Moreover, γ either remains a trail or becomes a pair of external trails after the unfolding, the latter only if γ is internal.*

Proof. It is sufficient to prove that the unfolding of just one shortcut $e = (v_1, v_2)$ in some elementary component C into a C-loop or C-fork α does not create an intersection with the rest of γ, and that the unfolding of γ has the desired properties.

(i) G is closed. Assume, by contradiction, that α intersects with γ. Setting out on γ from v_1 or v_2 in a positive M-alternating way (i.e. on an edge belonging to M) we must encounter a vertex that lies on α. Let u be the first such vertex, starting out from say v_1. On the interval from v_1 to u there is a last vertex v at which γ leaves component C. Making the appropriate shortcuts in C on the interval of γ from v_1 to v results in a positive M-alternating

path connecting these two vertices in C_h, indicating that $v_1 \not\sim_{C_h} v$. A contradiction is now immediate by Lemma 4.4.9. Obviously, γ is a single trail after the unfolding.

(ii) G is open. Consider the graph $(G^*)_h$ and the elementary component C^* in G^* containing C. In this setting γ determines an alternating trail γ^* in $(G^*)_h$, and α determines an alternating C^*-loop α^* with respect to any extension of M to a perfect matching. (See Theorem 4.3.7, Lemma 4.4.6, and Corollary 4.4.8.) Knowing from *(i)* that α^* and γ^* do not intersect, it follows that their subtrails α and γ do not intersect either. If α is a fork, then γ cannot be external, because in that case one of the two trails arising from the unfolding would be an inter-elementary crossing. This observation proves the second statement of the theorem. ♢

Corollary 4.4.15 *For any state $M \in S(G)$, an internal vertex w is M-accessible from external vertex v in G iff w is M-accessible from v in G_h.*

Proof. According to the last statement of the Shortcut Principle, by unfolding a hidden edge of a positive external M-alternating path we will also obtain a positive M-alternating path connecting the same vertices. Now using the above observation, the proof is a straightforward induction argument on the number of hidden edges. ♢

Corollary 4.4.16 *An edge $e \in E(G)$ is impervious in G iff e is impervious in G_h.*

Proof. Immediate by Corollaries 3.2.4 and 4.4.15. ♢

Let $\mathcal{P}(G)$ denote the canonical partition of $\mathrm{Int}(G)$ determined by the equivalence $\sim$.

Corollary 4.4.17 $\mathcal{P}(G) = \mathcal{P}(G_h)$.

Proof. Immediate by Lemma 3.2.7and the Shortcut Principle. ♢

Let $\mathcal{F}(G)$ and $\mathcal{H}(G)$ denote the sets of forbidden and hidden edges of G.

Corollary 4.4.18 $\mathcal{F}(G_h) = \mathcal{F}(G) \cup \mathcal{H}(G)$.

Proof. For any graph G, the set of forbidden edges consists of:

a) the edges connecting two different elementary components in G;

b) the forbidden edges of the elementary components themselves.

By Corollary 4.4.13, edges in *a)* are common for G and G_h. Moreover, by Theorem 4.4.11, the forbidden edges of G_h belonging to *b)* are exactly the hidden edges of G. ♢

In the sequel, by a canonical class of some elementary component C we shall mean a class by the partition $\mathcal{P}(G) = \mathcal{P}(G_h)$, rather than one by the partition associated with the equivalence $\sim_C$. According to Corollary 4.4.12, $\mathcal{P}(G)$ is determined locally by the equivalence relations $\sim_{C_h}$.

4.5 Structuring the elementary components

In the previous section we were concerned with the behavior of one particular elementary component of G when placed in the global environment determined by the surrounding elementary components. In this section we look at the global environment itself, and investigate the structure of all elementary components in G. Elementary components will be related to each other according to their accessibility from external vertices by alternating paths. Unlike in the previous sections, we shall use the phrase "external alternating path γ *enters* elementary component C" in the strict sense, meaning that γ enters C for the first time. Obviously, the path γ must then be negative.

Definition 4.5.1 An elementary component of G is *viable* if it does not contain impervious allowed edges. A viable internal elementary component C is *one-way* with respect to some state M of G if all external M-alternating paths enter C in the same canonical class of C. This unique class is called *principal* in C. Further to this, every external elementary component is a priori one-way by the present definition (with no principal canonical class, of course). A viable elementary component is *two-way* if it is not one-way. An *impervious* elementary component is one that is not viable.

It is easy to see that an impervious elementary component consists of impervious edges only. On the contrary, let M be a state of G and assume that there exists a positive external M-alternating path α leading to some vertex of an impervious elementary component C. Let v be the vertex of C where α enters this component, and denote by β the prefix of α up to v. By Claim 4.2.2, C is internal. Moreover, if $e = (v_1, v_2)$ is an arbitrary allowed edge of C, then clearly at least one of v_1 and v_2, say v_1, attracts v. Consequently, there exists a positive $M_{(C_h)}$-alternating path γ in C_h connecting v and v_1. Thus, the positive external alternating path $\beta\gamma$, with the help of Corollary 4.4.16, proves e to be viable. Since e was arbitrary, this contradicts the fact that C is impervious.

Proposition 4.5.2 *The one-way property is matching invariant with the principal canonical class preserved.*

Proof. Consider a negative external alternating path γ entering C in state M, and let M' be any other state. As in the proof of Proposition 4.4.5, restrict G and M to the elementary components visited by γ, and designate the last edge of γ incident with C as an external edge. In the resulting graph G_γ, γ becomes an $M_{(\gamma)}$-alternating crossing. Make the crossing γ in state $S(M_{(\gamma)}, \gamma)$, and then switch to state $M'_{(\gamma)}$. Apply the argument under Proposition 4.4.5 to conclude that there exists an M'-alternating crossing γ' in G_γ with the same endpoints and visiting the same elementary components as γ. Thus, γ' determines an M'-alternating external path in G entering C at the very same vertex as γ. In this way we have shown that the entry points of external alternating paths in C are the same with respect to all states of G. $\diamondsuit$

Proposition 4.5.3 *Let C be a viable internal elementary component of G. Then a C-fork exists in any state M only if C is one-way, and the internal endpoints of the fork are in the principal canonical class of C. The corresponding hidden edge in C_h is impervious in G_h.*

Proof. Let (α_1, α_2) be an M-alternating C-fork in G connecting vertices v_1 and v_2 belonging to a canonical class P of C. Suppose, by contradiction, that there exists a negative external M-alternating path γ entering C_h in a vertex v belonging to a canonical class different from P. By Lemma 4.4.9, γ must avoid the fork (α_1, α_2). But then a crossing would be obtained in G_h through γ, a positive M-alternating path in C_h from v to v_1 (v_2) and α_1 (respectively, α_2). We conclude that C is one-way with the class P being principal. Observe that all vertices v in any principal canonical class P are inaccessible. Indeed, if there was a positive external alternating path γ leading to v, then $v \not\sim u$ would hold for the vertex u where γ enters C. This is impossible, however, since u is also in class P. The edge (v_1, v_2) is therefore impervious. $\diamondsuit$

Proposition 4.5.4 *Component C is one-way in G iff C_h is one-way in G_h, and the principal canonical class of C is the same as that of C_h.*

Proof. It is sufficient to prove that if C is one-way and internal, then C_h is also one-way and its principal canonical class is that of C. On the contrary, assume that C is one-way with principal canonical class P, yet, there exists a negative external alternating path γ in G_h with respect to some state M of G_h that enters C_h at a vertex v belonging to a class different from P. Using the Shortcut Principle (Theorem 4.4.14), let us unfold the hidden edges on γ one-by-one, starting from the external vertex, into pairwise disjoint M-alternating loops until an intersection occurs with C at some vertex u. This intersection will indeed occur, otherwise the unfolding of γ would enter C at vertex v. At the vertex u, an appropriate external subpath of the unfolding of γ enters C, therefore u is in class P. Extend γ by a positive M-alternating path inside C_h up to the vertex u to obtain an M-alternating path γ'. It is now obvious that the Shortcut Principle fails to work for γ', which is a contradiction. $\diamondsuit$

Definition 4.5.5 Component C' is *two-way accessible* from component C with respect to some state M, in notation $C\rho C'$, if C' is covered by an appropriate M-alternating C-loop α. It is required, though, that if C is one-way and internal, then the endpoints of α *not* be in the principal canonical class of C.

Let C' be two-way accessible from C via loop α. The endpoints of α in C are called the *domain* vertices of α, while the *range* vertices of α (on C') are the vertices at which α first hits C' from both ends. The common canonical class of the domain vertices in C is also called *domain* , and the classes of the range vertices in C' are called range *range* as well. Clearly, the two range classes are different. The two negative alternating paths connecting the domain and range vertices within α are called the *(C'-) branches* of α.

According to Definition 4.5.5, if C is internal and $C\rho C'$ via loop α, then there exists an external alternating path entering C in a vertex belonging to a canonical class different from the domain of α. This observation will often be used in the sequel.

Lemma 4.5.6 *If $C\rho C'$ with respect to M, then C' cannot be one-way.*

Proof. Let α be a C-loop covering C' from domain class P. Suppose first that C is viable. By Claim 4.2.2, at most one of C and C' can be external. If C' were external, then C, being internal, could be entered by an external M-alternating path γ in a vertex belonging to a canonical class different from P. By Lemma 4.4.9 and Corollary 4.4.10, γ avoids the loop α and component C', which contradicts Claim 4.2.2 again. We conclude that C' is internal. In this case, however, regardless of C being internal or external, C' can be entered by an external M-alternating path through C and the loop α in both range vertices of α, which proves that C' is two-way.

Now let C be impervious, and assume by way of contradiction that C' is viable, let alone one-way. Let γ be an external M-alternating path entering C' at some vertex u. Clearly, there exists a positive M-alternating path β connecting u with at least one of the range vertices of α inside C'. If γ does not intersect with α, then C could be entered through γ, β, and an appropriate branch of α, contradicting that C is impervious. The same contradiction arises if γ does overlap with α, since in this case one can simply switch from γ to α at the first overlap to reach C from one direction. $\diamondsuit$

Proposition 4.5.7 *The relation ρ is matching invariant.*

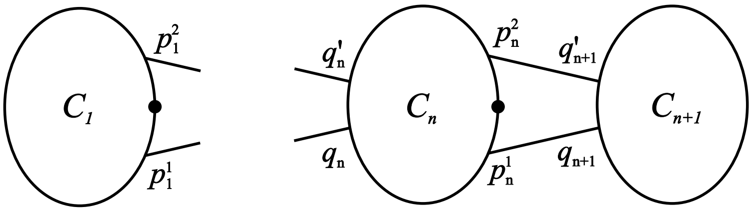

Figure 4.5: The proof of Lemma 4.5.8

.

Proof. By Definition 4.5.5, if $C\rho C'$ via some C-loop α, then $C\rho C''$ holds for every elementary component C'' covered by α. Lemma 4.5.6 then implies that all elementary components covered by α are internal. Now the statement follows directly from Proposition 4.4.5. $\diamondsuit$

Let us fix a state M for the rest of the section. All alternating paths, C-loops, etc., will be meant with respect to this state. Since all the concepts to be dealt with are matching invariant, the choice of M is irrelevant.

Lemma 4.5.8 *Let $C_1, \ldots, C_n$ $(n \geq 2)$ be elementary components such that $C_i \rho C_{i+1}$ for all $1 \leq i \leq n-1$ by appropriate C_i-loops α_i with domain vertices p_i^1, p_i^2 and range vertices q_i, q_i'.*

(i) The components $C_1, \ldots, C_n$ are all different.

(ii) For either choice $q \in \{q_n, q_n'\}$ there exists $j \in \{1,2\}$ such that p_1^j is connected to q by a negative alternating path β in G_h running essentially outside $C_1 \cup C_n$. Moreover, every edge of β is either on a loop α_i or belongs to some elementary component $(C_i)_h$.

(iii) If v is a vertex in C_1 such that $v \not\sim p_1^j$ $(j = 1,2)$, then there exists no alternating path β in G_h running essentially outside C_1 and connecting v with any vertex in C_n.

Proof. Induction on n. For $n = 2$ statements (i) and (ii) are straightforward, while (iii) is equivalent to Corollary 4.4.10. Assume that all three statements hold for some $n \geq 2$, and proceed to $n+1$. See Fig. 4.5 for an illustration.

(i) Assume, by contradiction, that $C_{n+1} = C_m$ for some $1 \leq m \leq n$. Without loss of generality we can take $m = 1$. Then at least one C_1-branch of α_{n+1} violates part (iii) of the induction hypothesis when that branch is taken for β.

(ii) By (i) above we already know that C_{n+1} is different from all C_i, $1 \leq i \leq n$. Choose $q \in \{q_{n+1}, q_{n+1}'\}$ arbitrarily, and let q be connected to p_n^k by the branch α_n^k of α_n, where $k \in \{1,2\}$. Since $q_n \not\sim q_n'$ holds in $(C_n)_h$, either $q_n \not\sim p_n^k$ or $q_n' \not\sim p_n^k$. Say $q_n \not\sim p_n^k$. Then there exists a positive alternating path β' in $(C_n)_h$ between p_n^k and q_n. On the other hand, the induction hypothesis provides an appropriate negative alternating path β_j between p_1^j and q_n for some $j \in \{1,2\}$, and Lemma 4.4.9 ensures that β_j does not overlap with α_n^k. Moreover, Corollary 4.4.10 ensures that β_j does not reach C_{n+1} either. In this way $\beta_j \beta' \alpha_n^k$ becomes a negative alternating path, which connects p_1^j with q in G_h, running essentially outside $C_1 \cup C_{n+1}$ with the desired edge composition.

(iii) Contrary to the statement, assume that an undesired alternating path β exists. By the induction hypothesis, β connects v with a vertex u in C_{n+1} in such a way that it avoids all the components C_i, $1 \leq i \leq n$, and loops α_i, $1 \leq i \leq n-1$. Without loss of generality we can also assume that β runs essentially outside C_{n+1}, so that it is at vertex u where β first hits any elementary component along the loop α_n. Clearly, $u \not\sim q_{n+1}$ or $u \not\sim q_{n+1}'$ holds in C_{n+1}, say $u \not\sim q_{n+1}$. A big alternating unit will then show up in G_h containing β, a positive alternating path in $(C_{n+1})_h$ connecting u with q_{n+1}, a positive alternating path or fork in $(C_1)_h$

connecting v with p_1^1 (p_1^2), and a negative alternating path connecting q with p_1^1 (respectively, p_1^2) according to (ii). $\diamondsuit$

Corollary 4.5.9 *With the parameters of Lemma 4.5.8, if v is an arbitrary vertex in C_n, then there exists an alternating path β in G_h connecting v with one of p_1^1 and p_1^2 in such a way that*

(a) β is positive at the v end and negative at the other end;

(b) every edge of β is either on a loop α_i, $1 \leq i \leq n-1$, or belongs to $(C_i)_h$ for some $2 \leq i \leq n$.

Proof. Since $v \not\sim q_n$ or $v \not\sim q'_n$, the statement follows directly from Lemma 4.5.9(ii). $\diamondsuit$

Corollary 4.5.10 *The transitive closure of ρ is asymmetric.*

Proof. Immediate by Lemma 4.5.9(i). $\diamondsuit$

Corollary 4.5.11 *The connection $C\rho C'$ holds in G iff $C_h\rho C'_h$ holds in G_h.*

Proof. It is sufficient to prove that $C_h\rho C'_h$ implies $C\rho C'$. Let α_h be a C_h-loop covering C'_h, and unfold α_h using the Shortcut Principle. By Lemma 4.5.6, none of the components covered by α_h are one-way, and by definition, the loop α_h itself cannot be a single hidden edge connecting two vertices belonging to the principal canonical class of a one-way component either. Therefore, by Proposition 4.5.3, α_h unfolds into a trail α. We claim that α is a C-loop, and therefore $C\rho C'$. To this end we need to verify that α avoids C. Should α overlap with C, there would be a component D along α such that $C\rho D$ and $D\rho C$, which contradicts Corollary 4.5.10. $\diamondsuit$

Lemma 4.5.12 *For every two-way C' there exists a viable C such that $C\rho C'$.*

Proof. Assuming that C' is two-way, let γ_1 and γ_2 be two external alternating paths entering C' in different canonical classes. Clearly, γ_1 and γ_2 must overlap. If e is the last overlapping allowed edge along γ_1 and γ_2, then it is easy to see that $C_h\rho C'_h$ holds for the elementary component C containing e. Thus, by Corollary 4.5.11 , $C\rho C'$. $\diamondsuit$

Let ρ^* denote the reflexive and transitive closure of ρ. By Corollary 4.5.10, ρ^* is a partial order.

Lemma 4.5.13 *Let C_1 and C_2 be two different elementary components of G such that $C_1\rho^*C$ and $C_2\rho^*C$ for some elementary component C. Then C_1 and C_2 cannot both be one-way.*

Proof. Based on Proposition 4.5.4 and Corollary 4.5.11 we can change the present setting from graph G to graph G_h. Let $C_1 = C_1^1\rho C_1^2\rho\ldots\rho C_1^n = C$ and $C_2 = C_2^1\rho C_2^2\rho\ldots C_2^m = C$ for appropriate components C_j^i, $j = 1, 2$, $1 \leq i \leq n$ (m) via some loops α_j^i. By Lemma 4.5.8 (ii) there exists a negative alternating path β_1 connecting a domain vertex v_1 in C_1^1 with a range vertex v_n in C_1^n, running essentially outside $C_1 \cup C_1^n$ with an appropriate edge composition. If β_1 covers C_2, then $C_1\rho^*C_2$, therefore C_2 is not one-way by Lemma 4.5.6. Otherwise follow β_1 starting from v_1, and let C' be the first among those elementary components covered by β_1 that are also covered by some of the C_2^i-loops α_2^i. Note that C' exists, as C is always a candidate to be chosen for C' at last. Clearly, $C_2\rho^*C'$ via the column of loops $\alpha_2^1, \ldots, \alpha_2^i$ for some $1 \leq i \leq m-1$. Let v be the vertex in C' where β_1 enters this component, and let β'_1 denote the subpath of β_1 from v_1 to v. Apply Corollary 4.5.9 to obtain an alternating path β_2 connecting v with a domain vertex v_2 in C_2, so that β_2 is positive on the v end and negative on the v_2 end. By the choice of C', $\beta = \beta_1'\beta_2$ is a negative alternating path between v_1 and v_2 running essentially outside $C_1 \cup C_2$. We shall make use of the path β in the next paragraph.

Let the vertices v_1 and v_2 belong to canonical classes P_1 and P_2, and assume by way of contradiction that both C_1 and C_2 are one-way. According to Claim 4.2.2, one of C_1 and C_2, say C_1, is internal. Then there exists an external alternating path γ_1 entering C_1 at some vertex u_1 belonging to its principal class R_1. Clearly, $P_1 \neq R_1$ and P_2 is not principal either, for P_j, $j = 1, 2$ are the domain classes of the C_j-loops α_j^1. Without loss of generality we can assume that γ_1 does not reach C_2. Indeed, if γ_1 reached C_2, then C_2 would also be internal and we could continue the proof with C_2 and the prefix of γ_1 that enters C_2. If γ_1 and β overlap, then it is straightforward to assemble an external alternating path from parts of γ_1 and β which enters C_1 or C_2 in the non-principal canonical class P_1 (respectively, P_2). This contradicts both of these components being one-way. Assume therefore that γ_1 and β are edge-disjoint. Then γ_1, a suitable positive alternating path in C between u_1 and v_1, and β will form an external alternating path entering C_2 in class P_2, which is again a contradiction. $\diamondsuit$

Lemma 4.5.14 *Let C be one-way, and suppose that $C\rho^*C'$. Then every external alternating path entering C' must enter C first.*

Proof. Let $C = C_1\rho\ldots\rho C_n = C'$ via a column of C_i-loops α_i, $1 \leq i \leq n-1$. Contrary to the statement of the lemma, assume that there exists an external alternating path γ entering C' at vertex v without having visited C first. Without loss of generality we can assume that γ does not overlap with any of the loops α_i. But then it is possible to enter C at a domain vertex of α_1 through γ and an appropriate continuation from v that is available by Corollary 4.5.9. This is a contradiction, since the canonical class of any domain vertex is not supposed to be principal. $\diamondsuit$

Definition 4.5.15 A *family* of elementary components in G is a block of the partition determined by the equivalence relation $(\rho \cup \rho^{-1})^*$. A family $\mathcal{F}$ is *viable* if every elementary component in $\mathcal{F}$ is such. An *impervious* family is one that is not viable.

As we observed in the proof of Lemma 4.5.6, for elementary components C and C' such that $C\rho C'$, C is viable iff C' is viable. Thus, any impervious family will consist of impervious elementary components only.

Theorem 4.5.16 *Every viable family contains a unique one-way elementary component, called the root of the family. Every member of the family is only accessible through the vertices belonging to the principal canonical class of the root by external alternating paths.*

Proof. By Corollary 4.5.10 and Lemma 4.5.12, each viable family does contain a one-way elementary component. Let C_1 and C_2 be one-way elementary components in a family $\mathcal{F}$. By Lemma 4.5.6, there is no elementary component D in $\mathcal{F}$ such that $D\rho C_i$ for either $i = 1$ or $i = 2$. Thus, there exists $D \in \mathcal{F}$ such that $C_1\rho^*D$ and $C_2\rho^*D$. Lemma 4.5.13 then implies that $C_1 = C_2$. The second statement of the theorem is equivalent to Lemma 4.5.14. $\diamondsuit$

A family $\mathcal{F}$ is called *external* (respectively, *internal*) if its root, denoted by $r(\mathcal{F})$, is such, and *degenerate* if it consists of a single degenerate elementary component. A vertex v is said to be *principal* in viable family $\mathcal{F}$, if v belongs to the principal canonical class of $r(\mathcal{F})$.

Definition 4.5.17 Let C be the root of a viable family $\mathcal{F}$ and P be a non-principal canonical class of C. Then the *P-subfamily* of $\mathcal{F}$ is the set of two-way components C' for which there exists an elementary component C_1 such that $C\rho C_1$ via a loop with domain P and $C_1\rho^*C'$.

Proposition 4.5.18 *Let $\mathcal{F}$ be a viable family and $C = r(\mathcal{F})$. Then the following two statements hold.*

(a) If P_1 and P_2 are distinct non-principal canonical classes of C, then the P_1-subfamily and the P_2-subfamily are disjoint.

(b) If e is an edge connecting distinct elementary components C_1 and C_2 in $\mathcal{F}$ such that C_2 is a member of the P-subfamily for some non-principal canonical class P of C, then either C_1 is also contained in the P-subfamily, or $C_1 = C$ with the endpoint of e in C belonging to P.

Proof. We will prove the following statement, from which both claims of the Proposition directly follow.

Let $C_1^1, \ldots, C_n^1$ and $C_1^2, \ldots, C_m^2$ $(n, m \geq 2)$ be elementary components such that $C_1^1 = C_1^2 = C$, $C_i^1 \rho C_{i+1}^1$ for all $i \in [n-1]$ by appropriate C_i^1-loops α_i^1, and $C_j^2 \rho C_{j+1}^2$ for all $j \in [m-1]$ by appropriate C_j^2-loops α_j^2. Furthermore assume that the domain class of α_1^k is P_k for $k = 1, 2$, and there exists an edge $e = (v_1, v_2)$ such that $v_1 \in V(C_i^1)$, $v_2 \in V(C_j^2)$ for some $i \in [n]$ and $j \in [m]$. Then $i = j = 1$ holds, i.e. e is an edge in C.

In order to prove the above claim, suppose on the contrary that one of i and j, say i, is greater than 1, and let M be an arbitrary state of G. Then, based on Corollary 4.5.9, there exists an M-alternating path β in G_h connecting v_1 with a vertex p_1 of P_1 in such a way that it is positive at the v_1 end with running essentially outside C. By the above fact, $C_j^2 \neq C$, because otherwise $\beta + e$ would constitute an M-alternating C-loop having domain vertices in distinct canonical classes. Therefore, by the prefix β' of $\beta + e$ from p_1 to the first vertex belonging to an elementary component C_k^2 for some $2 \leq k \leq m$, we obtain a contradiction in Lemma 4.5.8, part *(iii)*. ♢

The following proposition describes the location of inaccessible vertices in viable families.

Proposition 4.5.19 *A vertex v of some viable family $\mathcal{F}$ is inaccessible iff v belongs to the principal canonical class of the root of $\mathcal{F}$.*

Proof. By Corollary 4.5.9, for every vertex v of a two-way elementary component C there exists an alternating path γ from a vertex u of the root to v inside $\mathcal{F}$ that is positive at the v end and negative at the u end. We also know from the construction under Lemma 4.5.8 that u is a domain vertex (i.e. non-principal), therefore γ can be extended to a positive external alternating path leading to v. The same holds true if v is an internal vertex of the root, but belongs to a non-principal canonical class. ♢

In the sequel we establish a partial order among the families. For this goal we need to characterize forbidden edges connecting distinct families.

Proposition 4.5.20 *Let e be a viable forbidden edge of G connecting two different families $\mathcal{F}_1$ and $\mathcal{F}_2$. Then both $\mathcal{F}_1$ and $\mathcal{F}_2$ are viable, and exactly one endpoint of e belongs to the principal canonical class of the root of either $\mathcal{F}_1$ or $\mathcal{F}_2$.*

Proof. By definition, at least one endpoint of the viable edge e is accessible, thus falls into a viable family. Since this endpoint is not principal by Proposition 4.5.19, the other endpoint also marks a viable family, even if that endpoint is principal (as we wish to prove). Suppose now, by contradiction, that both endpoints v_1 and v_2 of e are non-principal, and let α_1 and α_2 be positive external alternating paths leading to v_1 and v_2, respectively. The paths α_1 and α_2 must overlap, so that there exists a C-loop α for an appropriate elementary component C, which loop contains e. This is a contradiction, for the endpoints of e are in different families. ♢

If e is a viable edge connecting families $\mathcal{F}_1$ and $\mathcal{F}_2$, then we write $e : \mathcal{F}_1 \mapsto \mathcal{F}_2$ to indicate that the principal endpoint of e is in $\mathcal{F}_2$.

Lemma 4.5.21 *Let $e_1 : \mathcal{F}_1 \mapsto \mathcal{F}_2, \ldots, e_n : \mathcal{F}_n \mapsto \mathcal{F}_{n+1}$ ($n \geq 1$) be viable edges among families $\mathcal{F}_i$, $1 \leq i \leq n+1$. Then $\mathcal{F}_1 \neq \mathcal{F}_{n+1}$.*

Proof. Assume, by contradiction, that $\mathcal{F}_{n+1} = \mathcal{F}_1$. Without loss of generality we can assume that the families $\mathcal{F}_1, \ldots, \mathcal{F}_n$ are all different. Then, using Corollary 4.5.9 and Proposition 4.5.20 , we can construct a negative alternating path γ in G_h starting from a vertex u of $r(\mathcal{F}_1)$, going through the edges $e_1, \ldots, e_{n-1}$ and families $\mathcal{F}_2, \ldots, \mathcal{F}_n$, and returning to a vertex v of $\mathcal{F}_1$ via e_n, so that γ runs essentially outside $r(\mathcal{F}_1)$. We know that the vertex v is principal in $r(\mathcal{F}_1)$, while u is not. The path γ can then be closed inside $r(\mathcal{F}_1)$ to an inter-elementary alternating cycle, which is a contradiction. $\diamondsuit$

By Lemma 4.5.21, if $e : \mathcal{F}_1 \mapsto \mathcal{F}_2$ for some families $\mathcal{F}_1$ and $\mathcal{F}_2$, then $e' : \mathcal{F}_1 \mapsto \mathcal{F}_2$ for all viable edges e' connecting $\mathcal{F}_1$ and $\mathcal{F}_2$. This establishes and justifies $\mapsto$ as a binary relation between viable families. Let $\overset{*}{\mapsto}$ denote the reflexive and transitive closure of $\mapsto$.

Theorem 4.5.22 *The relation $\overset{*}{\mapsto}$ is a partial order on the collection of all viable families of G, by which the external families are maximal elements.*

Proof. Immediate by Lemma 4.5.21. $\diamondsuit$

The following important corollary shows that relation $\mapsto$ reflects the order by which families are traversed by external alternating trails.

Corollary 4.5.23 *Let $e = (v_1, v_2)$ be an edge connecting viable families $\mathcal{F}_1$ and $\mathcal{F}_2$ such that v_1 (v_2) belongs to $\mathcal{F}_1$ (respectively, $\mathcal{F}_2$). Furthermore let α be an external alternating trail traversing e. Then $\mathcal{F}_1 \mapsto \mathcal{F}_2$ iff α reaches v_1 before v_2.*

Proof. As e is a negative edge in α, we obtain that the endpoint of e traversed first by α is accessible. Now the statement follows from Propositions 4.5.19 and 4.5.20, and from Lemma 4.5.21. $\diamondsuit$

Our closing result characterizes the relationship between the viable and impervious parts of G. It will be a straightforward consequence of the following proposition, which describes the location of impervious edges in G.

Proposition 4.5.24 *An edge $e = (v_1, v_2)$ is impervious if and only if either at least one of v_1 and v_2 is in an impervious family or v_i ($i = 1, 2$) belongs to a principal canonical class of the root of some viable family.*

Proof. By Proposition 4.5.19, every internal vertex v of a viable family $\mathcal{F}$, v is accessible if and only if v is not a principal vertex of $r(\mathcal{F})$. Consequently, any edge e connecting vertices of viable families is impervious iff both of its endpoints are principal vertices. We have also seen earlier that an impervious family $\mathcal{F}$ consists of impervious elementary components only, which implies that any edge having endpoint in $\mathcal{F}$ will be impervious. Hence the proof is complete. $\diamondsuit$

An impervious edge $e \in E(G)$ is called *principal impervious* if at least one of its endpoints belongs to the principal canonical class of the root of some viable family.

Corollary 4.5.25 *Removing the principal impervious edges from G disconnects the viable families from the impervious ones.*

Proof. Immediate by Proposition 4.5.24. $\diamondsuit$

4.6 Characterizing families by factor-critical graphs and splitters

In this section we focus on families that are essentially soliton graphs by themselves.

Definition 4.6.1 A soliton graph G is *self-contained* if G is viable and it contains a unique non-degenerate family.

By the results of Section 4.5, a self-contained soliton graph is either one nondegenerate external family by itself, or it consists of a single internal family $\mathcal{F}$ and a number of degenerate external families (i.e. external vertices) connected to some of the principal vertices in $\mathcal{F}$ by forbidden edges. In the former case G is called *external*, while in the latter G is *internal*.

First we characterize bipartite self-contained graphs. For this goal we introduce the concept of *essentially* 1*-extendable* graphs, by which we mean internal self-contained soliton graphs having no forbidden internal edges.

Proposition 4.6.2 *A bipartite soliton graph G is self-contained iff G is either* 1*-extendable or essentially* 1*-extendable.*

Proof. It is enough to prove the 'Only If' part. To this end let G be a bipartite self-contained soliton graph, C be the unique one-way component of G and M be a state of G. Suppose by way of contradiction that G contains an M-alternating C-loop α with domain vertices v_1 and v_2 (forbidden edges in C can be considered as degenerate C-loops). We distinguish two cases.

Case 1: G is internal self-contained. Then let w_1 denote the vertex incident with v_1 by the edge covered by M. Since $v_2 \not\sim_C w_1$, there exists a positive internal $M_{(C_h)}$-alternating path β between v_2 and w_1. Now it is clear that $\alpha' = \alpha + \beta[v_2, v_1]$ is an M-alternating loop in G_h, thus applying the Shortcut Principle on α' we obtain a contradiction.

Case 2: G is external self-contained. In this case, according to Claim 4.2.1, for $i = 1, 2$ there exists a positive external $M_{(C)}$-alternating path γ_i with endpoint v_i. Observe that γ_1 must be overlapped with γ_2, because otherwise (γ_1, γ_2) would form an alternating fork. Now starting from v_1 let v_1' denote the first vertex along γ_1 for which $v_1' \in V(\gamma_2)$. Clearly, $\gamma = \gamma_1^{-1}[v_1, v_1'] + \gamma_2[v_1', v_2]$ cannot form a positive internal alternating path, consequently $\gamma + \alpha$ is an M-alternating loop, which is a contradiction again. ◊

In general case external self-contained graphs can be characterized by combining Propositions 3.2.5 and 4.5.19, which results in the following statement.

Proposition 4.6.3 *A soliton graph G is external self-contained iff G is factor-critical.*

Corollary 4.6.4 *A bipartite soliton graph G is factor-critical iff G is* 1*-extendable.*

Proof. Immediate by Propositions 4.6.2 and 4.6.3. ◊

Now we turn to characterizing internal self-contained graphs by complete splitters. Let G be a soliton graph, fixed for the rest of this section.

Proposition 4.6.5 *Every maximal splitter is the union of certain canonical classes in G.*

Proof. Let S be a maximal splitter in G. We have to show that if $u \in S$ with $u \in Int(G)$ being in a canonical class P of some elementary component C, then every vertex v of P is in S. To this end it is sufficient to prove that, for every $x \in Int(G)$, u attracts x iff v attracts x. By Corollary 4.3.5, we can consider the closure G^* of G during the proof. If x is also in component C, then we are through, as relation $\sim$ is an equivalence. If x is in component $D \neq C$ and u

attracts x, then by Lemma 3.2.7 there exists a positive alternating path α connecting u and x with respect to any state of G. Let z be the first vertex along α, starting from x, that is in component C. Clearly, $u \not\sim z$, so that $v \not\sim z$. But then there exists a positive alternating path β connecting v with z inside C_h. Applying the Shortcut principal for $\alpha[x, z]\beta$ then results in a positive alternating path connecting v with x, showing that v attracts x. Since the role of u and v is interchangeble, the proof is complete. $\diamondsuit$

Definition 4.6.6 A splitter S of G is *complete* if every $x \in S$ attracts all vertices in $Int(G)\backslash S$.

Lemma 4.6.7 *Let u be a principal vertex of some viable family $\mathcal{F}$ in G and let α be an alternating path connecting u with an internal vertex v such that α starts out from u on a positive edge. Then the following statements hold.*

- *(i) For every family $\mathcal{F}'$ reached by α, $\mathcal{F} \overset{*}{\mapsto} \mathcal{F}'$.*
- *(ii) If α terminates in a positive edge at the endpoint v, then v is not principal.*

Proof. Induction on the number of families reached by α. If α stays within $\mathcal{F}$, then both statements are obvious. For the induction step, starting from u, assume that α leaves $\mathcal{F}$ at some vertex w to enter another family $\mathcal{H}$ at vertex x. Since the edge (w, x) is negative, u attracts w. The vertex w therefore cannot be principal. Proposition 4.5.20 then implies that x is principal in $\mathcal{H}$. The statements of the lemma now follow from a straightforward induction argument. $\diamondsuit$

Corollary 4.6.8 *Let S be a complete splitter of G consisting of inaccessible vertices only. Then G is internal self-contained with S being the principal canonical class of its unique internal family.*

Proof. First observe that S contains of all principal vertices of G. Indeed, by Lemma 4.6.7, part (ii), S either contains all of these vertices, or it contains none of them. In the latter case, by Proposition 4.5.19, all vertices of S would be in impervious components. This is impossible, however, since by Lemma 4.6.7, no vertex of an impervious elementary component can attract either a principal vertex in an internal family or any vertex in an external family.

Now assume that G has an impervious elementary component C. Clearly, all vertices of C cannot be in S, so there exists a vertex v in C that is attracted by all principal vertices of G. This, however, contradicts Lemma 4.6.7, part (i). The graph G therefore consists of viable elementary components only.

Choose $u \in S$ in an arbitrary way, and assume that u belongs to family $\mathcal{F}$. By Proposition 4.5.19, $\mathcal{F}$ is internal and u is principal in $\mathcal{F}$. Moreover, G cannot have a non-degenerate family $\mathcal{H}$ such that $\mathcal{F} \overset{*}{\not\mapsto} \mathcal{H}$. Indeed, if $\mathcal{H}$ were such a family, then it would contain a non-principal vertex v. Since v is accessible, $v \notin S$. The splitter S being complete, $u \in S$ would attract v. But this is again impossible by Lemma 4.6.7, part (i) .

Let us now assume that there exists a family $\mathcal{H} \neq \mathcal{F}$ such that $\mathcal{F} \overset{*}{\mapsto} \mathcal{H}$. Since none of the principal vertices of $\mathcal{H}$ can attract u, these vertices must belong to S. This, however, contradicts the observation in the previous paragraph when interchanging the role of $\mathcal{F}$ and $\mathcal{H}$. We conclude that G is self-contained, consisting essentially of the family $\mathcal{F}$ alone. It is also clear that S must be the principal canonical class of $\mathcal{F}$. $\diamondsuit$

The converse of Corollary 4.6.8 is also true.

Proposition 4.6.9 *The principal canonical class of every internal self-contained graph G is a complete splitter in G.*

Proof. Let P be the principal canonical class of G, and $u \in P$ be arbitrary. Consider the graph G_u that is obtained from G by first deleting all of its external edges, then attaching a single external edge to vertex u. Clearly, G_u is still self-contained, consisting of the same internal family $\mathcal{F}$ as G. Thus, by Corollary 4.5.19, u attracts every non-principal vertex of $\mathcal{F}$ both in G and G_u. ◇

Theorem 4.6.10 *A soliton graph G is self-contained iff it is factor-critical or its inaccessible vertices form a complete splitter.*

Proof. If G contains an internal family, then the statement of the theorem follows from Corollary 4.6.8 and Proposition 4.6.9. Otherwise the statement is equivalent to Proposition 4.6.3. ◇

4.7 Isolating the families of soliton graphs

In this section, making use of the structural results of this chapter, an algorithm is developed to isolate the families of a given soliton graph. For this end we need the following technical observations.

Let G be an internal self-contained soliton graph with $X = \{x_1, \ldots, x_k\}$ $(k \in N)$ being its principal canonical class. The set X is a complete splitter, consequently, by virtue of Theorem 3.3.3, every perfect internal matching M of G determines a sequence $C_1, \ldots, C_k$ of the odd connected components in $G - X$ such that x_i is connected to some vertex in C_i by an M-positive edge for every $1 \leq i \leq k$. Construct a new directed graph $G_X(M)$ over the vertices $x_1, \ldots, x_k$ in the following way. For every $1 \leq i \neq j \leq k$, draw an edge from x_i to x_j iff some vertex of C_i is adjacent to x_j in G.

Proposition 4.7.1 *The directed graph $G_X(M)$ is strongly connected for all states M.*

Proof. By Proposition 4.6.9, X is a complete splitter in G. Thus, every $x \in X$ attracts all vertices in $G - X$. Fix the numbers $1 \leq i \neq j \leq k$ in an arbitrary way, and construct a positive M-alternating path α connecting vertex x_i with any vertex of C_j in G. Clearly, α outlines a directed path in $G_X(M)$ from vertex x_i to vertex x_j, which proves the statement of the proposition. ◇

The algorithm to separate the viable families of a given soliton graph G is essentially a simplified version of the well-known Edmonds matching algorithm (cf. [40],[48]) with some modifications. The algorithm will assume that a perfect internal matching M has previously been found for G. According to Proposition 4.3.2, this can be achieved by applying any of the known matching algorithms for the closure G^* of G. (See e.g. [48], [50], and [86]) These algorithms run either in $\mathcal{O}(n \cdot m)$ or in $\mathcal{O}(\sqrt{n} \cdot m)$ time. However, the closure of G might have $\mathcal{O}(n^2)$ edges by which the running time will be $\mathcal{O}(n^3)$ or $\mathcal{O}(n^{5/2})$, respectively. Nevertheless, by careful implementation, these algorithms can be modified for soliton graphs in such a way that they can preserve the original complexity.([22]) Also note that a simple algorithm based on external alternating path methods (cf. [77]) can be developed directly for soliton graphs, which runs also in $\mathcal{O}(n \cdot m)$ time.

Our algorithm is broken down into two phases. Phase 1 builds up an M-alternating forest F to separate the inaccessible vertices from the accessible ones in G. At the same time, all external families of G are identified and the impervious elementary components are removed. Based on this information, Phase 2 isolates the internal families of G. Both phases have time complexity $\mathcal{O}(m)$, where m is the number of edges in G, not counting the loops and multiple edges.

Description of Phase 1

The forest F is built up by stepping on so called inner and outer vertices in an alternating fashion. Initially, F consists of the external vertices of G, so that $v \in Ext(G)$ is inner (outer) iff v is covered (respectively, not covered) by M. As the algorithm proceeds, vertices in F will be associated with groups of vertices in G. A vertex x in F is then called internal if all the vertices of G associated with x are internal, otherwise x is called external in F.

Every time an inner vertex u is found as a leaf of F, F is automatically extended by the unique edge $e \in M$ incident with u, and the other endpoint of e is classified as outer in F. For every M-negative edge e of G incident with an outer vertex v in F, one of the following three actions is considered.

1. *Augmentation.* If $e = (v, u)$ with $u \notin F$, then u is added to F as an inner vertex together with the edge e, which is marked as checked.

2. *Shrinking.* If $e = (v, u)$ with $u \in F$ being outer, then u and v are either in the same tree, or they belong to different trees in F. In the former case, find the closest common ancestor x of u and v, and consider the M-alternating loop α consisting of the unique paths in F from x to u and v plus the edge e. In the latter case, α is established in the same way as an M-alternating crossing. In either case, shrink α into one vertex v_α and classify v_α as outer in F. After the shrinking, all the M-negative edges of G incident with the outer vertices of α, different from the edges along α, are made available for consideration from v_α. The vertex v_α becomes external in F if α is a crossing.

3. *Null.* If $e = (v, u)$ with $u \in F$ being inner, then the action is void, and the edge e is marked half-checked.

Phase 1 stops if no unchecked M-negative edges are available for consideration at any outer vertex of F.

Concerning the time complexity of Phase 1, observe that the M-positive edges of G are considered at most once during this phase, while each edge $e \notin M$ is considered at most twice. Indeed, if $e = (u, v)$ first comes up under action 3 from vertex v and remains half-checked, then e will be considered the second time iff the vertex u gets involved in a shrinking action afterwards. Organizing the augmentation step in a depth-first manner will ensure that, whenever the edge e in 2 above initiates the shrinking of a loop, the two endpoints of e lie on the same branch of F. (In other words, e is a back edge for some tree in F.) Thus, shrinking will only take time proportional to the number of edges involved, not counting set operations. If we use the incremental tree disjoint set union algorithm ([49]), the set operations take a total of $\mathcal{O}(m)$ time as well. These arguments show that Phase 1 can be done in $\mathcal{O}(m)$ time.

With some extra administration we can list all the accessible vertices and isolate the external families of G in Phase 1. To this end we maintain a set variable *Accessible*, which contains the accessible vertices located during the algorithm. In addition, every time an M-alternating loop or crossing α is shrunk into the vertex v_α in action 2 above, a new set variable Acc_{v_α} is created and assigned the set of all vertices along α. If, later on, v_α becomes part of a shrinking action for some loop or crossing β, then v_α will contribute the whole contents of Acc_{v_α} to the contents of Acc_{v_β}, rather than just itself as an "artificial" vertex. As to the global variable *Accessible*, it is incremented by every new outer vertex that is added automatically to F after an augmentation action, and by the internal contents of the variables Acc_{v_α} every time a shrinking action is performed.

Theorem 4.7.2 *After the execution of Phase 1, the contents of the variable Accessible is the set of all accessible vertices in G. External vertices of F correspond to external families of G*

in a one-to-one manner, and for each external vertex x of F, the contents of Acc_x is the set of all vertices of G belonging to the family that corresponds to x.

Proof. Clearly, every vertex added to *Accessible* at any time of the algorithm is accessible in G. Conversely, let β be any positive external M-alternating path leading to an internal vertex v in G. We prove by induction on the number of positive edges in β that v is in *Accessible*. The basis case, when β contains a single positive edge, is obvious. Suppose that the statement is true for all appropriate paths β containing n positive edges for some $n \geq 1$, and let γ be an external alternating path to some internal vertex u containing $n+1$ positive edges.

Let v be the endpoint of the prefix β of γ up to the nth positive edge. Furthermore, let e and f denote the last two edges of γ with x being the common endpoint of e and f. By the induction hypothesis, $v \in$*Accessible*. Consider the instance of F when v was added to *Accessible*. No matter how this happened, v appeared as (part of) an outer vertex of F at that time. Assuming that x was not in F at the same time, the algorithm could either augment F by the edges e and f immediately, or wait until x became an outer vertex in F and then perform a shrinking action involving all three vertices v, x, u. Either way, u got added to *Accessible*.

Assume now that x was already in F when v was added to *Accessible*. If x was inner, then u was also in F as outer, so included in *Accessible*. If, however, x was outer, or merged into an outer vertex, then shrinking was applicable for the edge e, and, as a consequence, u got added to *Accessible*.

Looking at the final form of F, it is clear that the inaccessible vertices of the viable part of G appear as the internal inner vertices of F. Moreover, every external family $\mathcal{F}$ of G is shrunk into one external vertex in F. If $\mathcal{F}$ is degenerate, then $\mathcal{F}$ itself is an inner external vertex of F. Otherwise notice that $\mathcal{F}$ will be shrunk into one outer vertex in F. Indeed, every internal vertex u of $\mathcal{F}$ is accessible by an external alternating path α running inside $\mathcal{F}$. Therefore, if u was shrunk into an inner vertex of F, then some edge e of α would connect two distinct outer vertices of F. However, this is not possible, because in that case e must have been shrunk into an outer vertex by applying action 2. Now summarizing the above facts, we obtain the second statement of the theorem. $\diamondsuit$

Description of Phase 2

Phase 2 of the algorithm starts from the graph $\bar{G}$, which is obtained from G by deleting the impervious part of G and shrinking each external family into a single external vertex. This graph can easily be constructed as a by-product of Phase 1. A new M-alternating forest $\bar{F}$ is then built in the same way as in Phase 1, starting now from the inaccessible vertices of $\bar{G}$. Each of these vertices is classified as inner. Note that in this phase the shrinking action is applicable for alternating loops only, because no positive alternating path exists between two principal vertices. (see Lemma 4.6.7)

Let $X = \{x_1, \ldots, x_k\}$ be the set of all inaccessible vertices in $\bar{G}$ and $Y = \{y_1, \ldots, y_l\}$ be the set of all external vertices in $\bar{G}$. In the process of building up $\bar{F}$, a new directed graph $\bar{G}_{X,Y}$ is constructed over the vertices $X \cup Y$ in the following way.

1. For every edge (y_i, x_j) in $\bar{G}$, put (y_i, x_j) in $\bar{G}_{X,Y}$.

2. Draw an edge from x_i to x_j $(i \neq j)$ in $\bar{G}_{X,Y}$ iff the tree of $\bar{F}$ rooted at x_i contains an outer vertex adjacent to x_j in $\bar{G}$.

Decompose the graph $\bar{G}_{X,Y}$ into strong components, and make the following synthesis. To every strong component $H \subseteq X$ there corresponds a viable internal family of G, which is recovered by joining the vertices visited during the construction of the trees of $\bar{F}$ rooted at vertices in H. The partial order $\overset{*}{\mapsto}$ of G's families is that of the strong components of the graph $\bar{G}_{X,Y}$.

Theorem 4.7.3 *Phase 2 correctly identifies the internal families and determines the partial order* $\stackrel{*}{\mapsto}$ *of all viable families in* G.

Proof. By Lemma 4.6.7) and Proposition 4.5.19, X is a maximal inaccessible splitter in G^+, thus Theorem 3.4.5 implies that $c^{\rm o}_{\rm in}(G^+, X) = |X|$. Then applying Theorem 4.5.25 we obtain that the graph $G - X$ also has k odd internal connected components, which are spanned by the vertices visited during the construction of the k trees of $\bar{F}$, except for the vertices X themselves. Clearly, each of these trees will eventually reduce to one edge connecting an inner vertex in X with an outer vertex representing a component of $G - X$. Moreover, following the construction of $\bar{G}_{X,Y}$, it is easy to check that if vertices x_i and x_j belong to distinct families, then they will be in distinct strong components of $\bar{G}_{X,Y}$. Now the statement of the theorem follows from Proposition 4.7.1. $\diamondsuit$

Finding the strong components of a directed graph and determining the partial order of these components can be done in a time proportional to the number of edges in that graph (See e.g. [4], [72] or [93]). Consequently, the time complexity of Phase 2 is also $\mathcal{O}(m)$.

Chapter 5

Decomposition of soliton automata

5.1 Introduction

Concerning soliton circuits and soliton automata two questions seem to be the most fundamental to address.

(a) Given the underlying topology of interconnected molecules and molecule chains, verify the soliton circuit based on this system by describing its operations. (see e.g. [59])

(b) Characterize the class of soliton automata.

In this section, we will reduce both of the above problems to the the analysis of elementary soliton automata. The key result is stated in Sectionc5.4, where a decomposition is given on the ground of the structure theory developed in Chapter 4.

Translating question (a) into the language of soliton automata, it means that a method which describes the automaton associated with a given soliton graph (Automaton Description Problem - ADP) is needed. Naturally, the basic approach would suggest that one should construct the automata with all of its states and the transition function (Automaton Construction Problem - ACP). However, it is not clear at this moment whether ACP is even effectively computable. Indeed, the structure of the underlying graph is finite, which makes it possible to list all of its states, but the number of possible soliton walks is infinite, showing that the transition between two given states cannot be decided purely by definitions. Moreover, even if the existence of an algorithm is guaranteed, the complexity issues are still in question. The solution for ACP is based on two theorems of Section 5.2. The first one describes the soliton transitions between distinct states by alternating networks, while the second one characterizes the self-transitions – transitions from a state to itself – by external alternating trails. It turns out actually that a self-transition in a given state M is induced by a soliton walk starting from external vertex v iff there exists either an M-alternating v-loop or a so-called M-alternating double v-racket, by which we mean a pair of alternating v-rackets such that neither handle is the prefix of the other one. These results lead then to a polynomial time algorithm which can decide if a transition between two given states exists.

Problem (b) is the most fundamental theoretical question in the analysis of soliton automata. As we have seen in Chapter 2, the first results in the topic solved the problem only for special deterministic automata. Here, based on the structure theory developed in Chapter4, we will work out a decomposition of soliton automata by which the general problem is reduced to the analysis of soliton automata associated with elementary graphs. The first step towards the above goal is made in Section 5.3, which is devoted to the characterization of soliton automata with a single external vertex. We have two main reasons for investigating such automata: On the one hand, we generalize the result of [35] which provided a characterization of deterministic soliton automata with single external vertex, and on the other hand, these automata will play a central role in the decomposition. For describing the main result we introduce the concept of full (semi-full) automata, by which we mean automata (respectively, automata without self-transitions) with a single input symbol such that it has a transition between any two states (respectively, between any two distinct states). Then we prove that a soliton automaton with a single external vertex is either full or semi-full, the later holds only if its underlying graph is a bipartite graph having no double v-rackets. The main challenge in the proof is to show that a bipartite graph with a single external vertex contains a double v-racket in one state iff it does in all states.

The actual decomposition is worked out in Section 5.4. The main result is described with the help of the concept of soliton isomorphism, by which we mean a strong isomorphism between the extensions of the given automata in which the empty soliton walk is also allowed. We also define a special two-level ε-product called canonical product. Intuitively, in a canonical product the automata on the second level are connected to the soliton automata on the first level, through their canonical classes, according to a canonical dependency, which is simply a mapping from the set of automata on the second level to the power set of canonical classes of soliton automata on the first level. State transition is induced in an automaton on the second level according to its "accessibility" from the first component of the input pair through a canonical class determined by the canonical dependency. Using this terminology we will prove that for any soliton automaton $\mathcal{A}(G)$, there exists a soliton isomorphism between $\mathcal{A}(G)$ and a canonical product from the system of soliton automata associated with the hidden external elementary components of G to a system of full automata. More exactly, in the above product the full automata are the ones which correspond to an internal elementary component C of G with respect to the number of states of C, and the canonical dependency is determined by the relation $\stackrel{*}{\mapsto}$. Finally, by verifying that the class of soliton automata is closed under canonical products with respect to full automata, we will prove that the class of soliton automata and the class of automata obtained by a canonical product from a system of elementary soliton automata to a system of full automata coincide up to soliton isomorphism.

The above result gives the opportunity for future theoretical research to concentrate on elementary soliton automata only. However, it also provides a nice reduction for solving ADP. Soliton automata with a single external vertex shows the best example that the descriptional complexity of a soliton automaton might be much lower than the constructional complexity, as they are fully described by giving the number of their states. Therefore, making use of the preceding results of this chapter, in Section 5.5 we work out a structure encoding for soliton graphs. Such an encoding of a soliton graph G consists of the followings: the hidden elementary components extended by some loops following a certain rule, the canonical partition of these components, the identifiers of the full automata correspond to the internal elementary components with respect to their number of states, and a relation between the canonical classes and the full automata. The above reduced structure will be equivalent to the automaton associated with G, but it provides a lower complexity for ADP. Since the elementary decomposition in Section 5.4 is given by soliton isomorphism, we need to complete the above result with describing the self-transitions on the level of external elementary components. Finally, we will give an efficient algorithm which construct the structure code of any soliton graph. The algorithm runs in polynomial time iff the state complexity of the internal elementary components can be determined in polynomial time.

We note that the results of Section 5.3 has not been published earlier. The paper [11] is a preliminary version of Sections 5.2, 5.4 and 5.5. The algorithms of this chapter are from [22] and have not been published either.

5.2 The Automaton Construction Problem for soliton graphs

In this section we will investigate the following fundamental problem.

Automaton Construction Problem (ACP): *Given a soliton graph G. Construct the automaton $\mathcal{A}(G)$ associated with G.*

In order to solve ACP we need to determine the set of states and the transition function of $\mathcal{A}(G)$. For this goal three questions are to be answered:

(a) How to determine the viable part of G?

(b) How to list all the states of G^+?

(c) How to decide if there exists a transition between two given states?

The first question is answered by the algorithm of Section 4.7 which runs in $\mathcal{O}(|E(G)|)$ time. The second problem can be solved by adopting the following extension of the method suggested in [69] for bipartite graphs with perfect matchings.

Let G be a soliton graph with one of its states M. (M can be found efficiently with the help of the closure, as it was outlined in Section 4.7). The algorithm will use the straightforward consequence of Theorem 2.3.1 that a perfect internal matching is not unique iff it contains an alternating unit.

The idea borrowed from [69] is to define the procedure *NEWSTATES(G',M',α',L')* for any nice subgraph G' of G, perfect internal matching M' of G', M'-alternating unit α' and perfect internal matching L' of $G \setminus V(G')$. It finds all the additional perfect internal matchings of G'. A state of G is obtained by adding the set of edges L to a perfect internal matching of G'. Note that *NEWSTATES* is invoked only when M' is not unique.

The method operates as follows.

Let $(x,y) \in M'$ be an internal edge of the M'-alternating unit α'. The states of G' fall into two disjoint categories:

(a) States which do not cover (x,y): Then $M'_e = S(M',\alpha')$ does not contain the edge (x,y). Let $G_e = G - (x,y)$. There exists additional perfect internal matchings which do not contain (x,y) if and only if there exits an M'_e-alternating unit α'_e in G_e. These states can be found by invoking *NEWSTATES(G_e,M'_e,α'_e,L')* recursively.

(b) States which cover (x,y): Let $M'_v = M' \setminus \{(x,y)\}$. Then there exists additional perfect internal matchings which contain the edge (x,y) if there exists an M'_v-alternating unit α'_v in $G_v = G' \setminus \{x,y\}$. These states can be found by invoking *NEWSTATES(G_v, M'_v,α'_v, L'_v)* recursively, where $L'_v = L' \cup \{(x,y)\}$.

By the above facts *NEWSTATES(G',M',α',L')* consists of the following actions:

(1) Choose an internal edge $(x,y) \in E(\alpha') \cap M'$ and let $M'_e = S(M',\alpha')$.

(2) Let $G_e = G - (x,y)$ and find an alternating unit α'_e with respect to M_e; if none exists then $\alpha'_e = nil$.

(3) Let $M'_v = M' \setminus \{(x,y)\}$, $G_v = G' \setminus \{x,y\}$, and find an alternating unit α'_v with respect to M'_v; if none exists then $\alpha'_v = nil$.

(4) Output $M'_e \cup L'$.

(5) If $\alpha'_e \neq nil$, then call *NEWSTATES(G_e,M'_e,α'_e,L')*.

(6) If $\alpha'_v \neq nil$, then call *NEWSTATES(G_v,M'_v,α'_v,$L' \cup \{(x,y)\}$)*.

Now, starting from M, all the other states can be listed by finding an M-alternating unit and call the procedure *NEWSTATES(G, M, α, $\emptyset$)*. Using the same proof as that of Lemma

2. in [69], we obtain that the algorithm lists all states of G, and not counting the action of searching an alternating unit, any additional state is obtained in $\mathcal{O}(|E(G)|)$ time.

Therefore the complexity of the algorithm depends on the efficiency of the method to find an alternating unit in G. By the algorithm of Section 4.7, a crossing can be found in $\mathcal{O}(|E(G)|)$ time. Moreover, Gabow, Kaplan, and Tarjan (cf. [51]) recently have given a method which tests the existence of an alternating cycle with respect to perfect matchings also in $\mathcal{O}(|E(G)|)$ time. In any state M of G, deleting the external edges not contained in M, we obtain a perfect matching which shows that the algorithm of [51] can be applied in a straightforward way with the same complexity. Now summarizing the above observations we can conclude the following.

Theorem 5.2.1 *For any soliton graph G, the set of states $S(G^+)$ of $\mathcal{A}(G)$ can be constructed in $\mathcal{O}(|E(G)| + |S(G^+)| \cdot |E(G^+)|)$ time.*

According to the above facts we are left to solve problem (c). Since the definition of soliton walks is not flexible enough for this goal, it is a central question to describe the matching structure of the soliton transitions. As a first step we establish the correspondence between alternating networks and soliton walks.

Definition 5.2.2 Let M be a state of soliton graph G and $v, w \in Ext(G)$. An *M-transition network* Γ *from v to w* is a nonempty M-alternating network such that all elements of Γ, except one crossing from v to w if $v \neq w$, are alternating cycles accessible from v in M.

Theorem 5.2.3 *Let M, M' be distinct states of soliton automaton $\mathcal{A}(G) = ((S(G^+), (X \times X), \delta)$, and Γ be the mediator alternating network between M and M'. Then for any pair of external vertices $(v, w) \in X \times X$, $M' \in \delta(M, (v, w))$ holds iff Γ is an M-transition network from v to w.*

Proof. *'Only if'* By assumption, there exists a soliton walk α with respect to M which runs from v to w, and $S(M, \alpha) = M'$. The mediator alternating network Γ consists of the edges having different status in M and M', thus all edges of the alternating cycles in Γ are obviously traversed by α, and, in the case of $v \neq w$, Γ contains a unique crossing between v and w. Now, taking into consideration the above facts, the claim follows from Proposition 2.4.4.

'If' Let us construct for each cycle β of Γ an M-alternating v-racket β' such that the length of the handle of β' is minimal. Let Γ' denote the set of the above v-rackets and, in the case of $v \neq w$, of the crossing of Γ.

We will show by an inductive argument on $|\Gamma'|$ that there exists a soliton walk α with respect to M such that $E(\alpha) \subseteq \bigcup(E(\beta') \mid \beta' \in \Gamma')$ and $S(M, \alpha) = S(M, \Gamma)$. In the basis step $\Gamma = \{\beta'\}$, where β' is either a crossing or a v-racket. Then β', respectively $\beta' + (\beta'_h)^{-1}$, is a suitable choice for α.

For the induction step, assume that $|\Gamma'| > 1$ and let γ denote the v-racket of Γ' having the longest handle. First we prove that $V(\gamma_c)$ is disjoint from $\bigcup(V(\gamma') \mid \gamma' \in \Gamma' \setminus \{\gamma\})$. To this end, suppose on the contrary that $V(\gamma') \cap V(\gamma_c) \neq \emptyset$ for some $\gamma' \in \Gamma' \setminus \{\gamma\}$. Obviously, γ' is a v-racket and γ'_c is disjoint from γ_c. Therefore an M-alternating v-racket γ^1 can be constructed such that $\gamma^1_c = \gamma_c$ and γ^1_h is the prefix of γ'_h from v to the first vertex common with γ_c. However, by the construction of the v-rackets of Γ', we obtain that $|E(\gamma_h)| \leq |E(\gamma^1_h)|$, which implies the contradictory fact $|E(\gamma'_h)| > |E(\gamma_h)|$ with respect to the choice of γ.

Now using the induction hypothesis, consider a soliton walk α' with $E(\alpha') \subseteq \bigcup(E(\gamma') \mid \gamma' \in \Gamma' \setminus \{\gamma\})$ and $S(M, \alpha') = S(M, \Gamma \setminus \{\gamma_c\})$. Furthermore, let v' denote the internal endpoint of γ_h, and starting from v, let w' denote the last vertex of α' for which $\alpha[v, w']$ is a prefix of γ_h. Then it is easy to see that $\gamma + \gamma_h^{-1}[v', w'] + \alpha'[w', w]$ will be a soliton walk with the required properties; which makes the proof complete. $\diamondsuit$

Now the above theorem can be used for distinct states M_1 and M_2 to decide if a transition exists from state M_1 to state M_2 by a given input (v_1, v_2). The answer is positive, if the mediator alternating network between M_1 and M_2 is an M_1-transition network from v_1 to v_2. As the mediator alternating network Γ is given by the symmetric difference of M_1 and M_2, i.e. the alternating units of Γ consist of those edges which are covered by exactly one of M_1 and M_2, this problem simply requests to decide if all the alternating cycles in Γ are M-accessible from v. The following consequence of the algorithm of Section 4.7 shows that the above question can be answered in linear time.

Proposition 5.2.4 *For any state M of a soliton graph G and any external vertex $v \in Ext(G)$, the set of internal vertices M-accessible from v can be constructed in $\mathcal{O}(|E(G)|)$ time.*

Proof. Let e denote the external edge incident with v, and construct the graph G' from G by deleting the external edges which are different from e and not covered by M. Then add a loop around each external vertex in G' which is different from v and let G_v denote the resulted graph. Now applying Theorem 4.7.2 for G_v, we obtain the claim. $\diamondsuit$

Therefore combining the above result with the observations after Theorem 5.2.3, we obtain the following.

Corollary 5.2.5 *Let M_1 and M_2 be distinct states of $\mathcal{A}(G) = (S(G^+), X \times X, \delta)$, and $v, w \in Ext(G)$. Then it can be decided in $\mathcal{O}(|E(G^+)|)$ time if $M_2 \in \delta(M_1, (v, w))$ holds.*

Having solved the problem of transitions between distinct states, now we turn to *self-transitions*, i.e. transitions from a state to itself.

Definition 5.2.6 Let M be a state of soliton graph G and $v \in Ext(G)$. An *M-alternating double v-racket* α is a pair of M-alternating v-rackets (α^1, α^2) such that $E(\alpha_h^1) \cap E(\alpha_c^2) = \emptyset$, $E(\alpha_h^2) \cap E(\alpha_c^1) = \emptyset$, and either $\alpha_c^1 = \alpha_c^2$ or $V(\alpha_c^1) \cap V(\alpha_c^2) = \emptyset$. The maximal common external subpath – denoted by α_h – of α_h^1 and of α_h^2 is called the *handle* of α, whereas the internal endpoint of α_h is referred to as the *branching vertex* of α.

Note that the handle of an alternating double v-racket is a positive external alternating path. Figure 5.1 presents simple examples for the above definition.

Theorem 5.2.7 *For any state M of soliton automaton $\mathcal{A}(G) = ((S(G^+), (X \times X), \delta)$ and for any external vertex $v \in X$ of G, $M \in \delta(M, (v, v))$ iff one of the following conditions holds:*

(i) G does not contain an M-alternating v-racket.

(ii) G contains an M-alternating v-loop.

(iii) G contains an M-alternating double v-racket.

Proof. For an M-alternating v-loop α it is easy to check that $\alpha' = \alpha_h + \alpha_c + \alpha_c + \alpha_h^{-1}$ is a soliton walk from v to itself such that $S(M, \alpha') = M$. Therefore we can suppose throughout the proof that G does not contain an M-alternating v-loop.

'Only if' Suppose first that there does no exist a soliton walk α with respect to M such that α starts and terminates at v. Observe that in this case condition (i) holds. Indeed, G does not contain v-loops by assumption, and an M-alternating v-racket α would imply the contradictory fact that $\alpha_h + \alpha_c + (\alpha_h)^{-1}$ is a soliton walk.

By the previous paragraph, we can assume that there exists a soliton walk $\alpha = v_0, e_1, v_1, \ldots, v_{k-1}, e_k, v_k$ $(k > 2)$ with respect to M for which $v_0 = v_k = v$ and $S(M, \alpha) = M$.

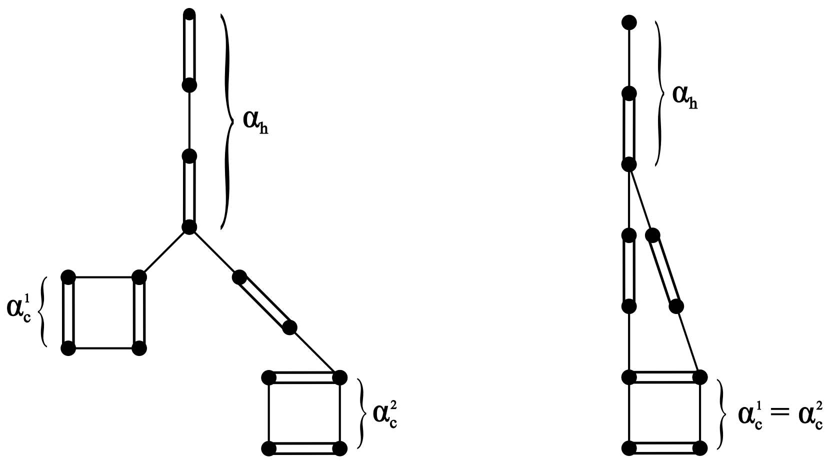

Figure 5.1: Example for double v-rackets

Then let $\alpha[v_0, v_i]$ ($i \in [k-1]$) be the shortest subpath of α with an index $j > i$ having the property that $v_i = v_j$, $n_\alpha(j) = 1$ and $\alpha[v_{i+1}, v_{j-1}]$ being vertex-disjoint from $\alpha[v_0, v_i]$. Note that the suitable indexes i, j exist, as α returns to itself along its way. As a first observation, we will prove that $e_{j+1} = e_i$.

For the above goal, notice that, based on Proposition 2.4.4 there exists an external M-alternating trail β terminating in e_j with $E(\beta) \subseteq E(\alpha[v_0, v_j])$. By assumption, β is necessarily an alternating v-racket, and because of the choice of v_i, $\beta_h = \alpha[v_0, v_i]$. By the above facts, $\alpha'' = \alpha[v_0, v_j] + \alpha^{-1}[v_i, v_0]$ is clearly a soliton walk with respect to M, implying that the edges traversed an odd number of times by $\alpha[v_i, v_j]$ will constitute an M-alternating network Γ consisting of alternating cycles only. As $\beta_h = \alpha[v_0, v_i]$, we obtain that $e_{i+1} \in M$; consequently, for the alternating cycle γ of Γ passing through e_j, $e_{i+1} \in E(\gamma)$. Therefore e_{i+1} is traversed by $\alpha[v_0, v_j]$ an odd number of times, which implies that $e_{j+1} \neq e_{i+1}$. Now we conclude that $e_{j+1} \notin M$ with $n(j+1)$ being even, which conditions are satisfied – because of the choice of v_i – only by e_i, as requested.

Clearly, as $S(M, \alpha) = M$, there exists an edge e_m of $\alpha[v_j, v_k]$ such that $e_m \notin E(\alpha[v_0, v_i])$, but for any index $j < l < m$, $e_l \in E(\alpha[v_0, v_i])$. Then let e_r denote the edge of $\alpha[v_0, v_i]$ for which $e_r = e_{m-1}$, and let s denote the smallest index with $m \leq s < k$ such that either $v_s \in V(\beta)$ or $v_s = v_l$ for some $m \leq l < s$. (See Figure 5.2) Again, note that a suitable index s exists, as e_j must be traversed by $\alpha[v_{m-1}, v_k]$. Furthermore, notice that $\alpha[v_0, v_{r-1}]$ and $\alpha[v_m, v_s]$ are vertex-disjoint. Indeed, if $v_l = v_p$ holds for some $0 \leq l \leq r-1$ and $m \leq p \leq s$ with $e_p \notin E(\alpha[v_0, v_{r-1}])$, then either $n_\alpha(p) = 1$ or $e_p \in E(\alpha[v_i, v_j])$, resulting in a contradiction with the choice of v_i.

It is clear that $e_m \notin M$ and for each edge e_l of $\alpha[v_j, v_{m-1}]$, $n_\alpha(l) = 2$ holds. Therefore $e_{m-1} \notin M$ and $n_\alpha(m)$ is odd, which, together with the observation of the preceding paragraph, implies that $\alpha' = \alpha[v_0, v_{r-1}] + \alpha[v_{m-1}, v_s]$ is a soliton walk with respect to M. Applying Proposition 2.4.4 again, there exists an external M-alternating trail γ terminating in e_s with $E(\gamma) \subseteq E(\alpha')$. Then a required double v-racket $\delta = (\beta, \gamma')$ can be constructed such that $\delta_h = \alpha[v_0, v_{r-1}]$, and either $\gamma' = \gamma$ (in the case of $v_s \notin V(\beta)$), or $\gamma'_h = \gamma$ with $\gamma'_c = \beta_c$.

'If' By assumption either (i) or (iii) holds for G. If G does not contain an M-alternating v-racket, then by Theorem 5.2.3 and by the definition of soliton automata, we obtain that $\delta(M, (v, v)) = \{M\}$, which proves the claim.

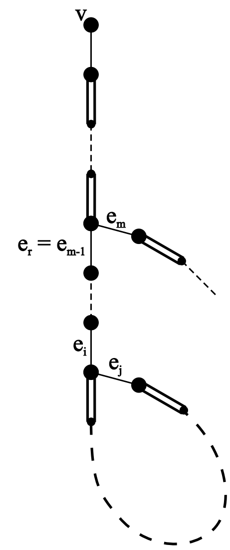

Figure 5.2: The proof of Theorem 5.2.7

Therefore consider an M-alternating double v-racket $\alpha = (\alpha^1, \alpha^2)$, and let w denote the branching vertex of α. Moreover, for $i = 1, 2$, let α_s^i denote the suffix of α_h^i from w to its internal endpoint, and let $\alpha_w^i = \alpha_s^i + \alpha_c^i + (\alpha_s^i)^{-1}$. If $\alpha_c^1 \neq \alpha_c^2$ – remember that in this case α_c^1 and α_c^2 are vertex-disjoint –, then $\alpha_h + \alpha_w^1 + \alpha_w^2 + \alpha_w^1 + \alpha_w^2 + \alpha_h^{-1}$ is a soliton walk with the desired properties. Otherwise, $\alpha_h + \alpha_w^1 + \alpha_w^2 + \alpha_h^{-1}$ will provide the requested walk. $\diamondsuit$

The following important corollary of the above theorems shows that any soliton automaton $\mathcal{A}(G)$ is strongly isomorphic with the automaton associated with the hidden graph of G.

Corollary 5.2.8 *Let $\mathcal{A}(G) = (S(G^+), X \times X, \delta)$ be a soliton automaton, $v, w \in X$ be external vertices and $M, M' \in S(G^+)$. Then $M' \in \delta(M, (v, w))$ iff $M' \in \delta_h(M, (v, w))$, where δ_h is the transition function of $\mathcal{A}(G_h)$.*

Proof. It is clear by Corollary 4.4.18 and by Theorem 2.3.2 that an alternating unit does not contain hidden edges, thus the alternating networks of G and that of G_h are the same. Furthermore, according to Corollary 4.4.15, an alternating cycle is M-accessible from v in G iff it is such in G_h. Therefore, for distinct states M and M', the statement immediately follows from Theorem 5.2.3.

Assume now that $M = M'$. Then, making use of the Shortcut Principle, it is easy to see that G contains an M-alternating v-racket iff G_h does. Moreover, making a shortcut on an M-alternating v-loop (or on an M-alternating double v-racket) will result in the same kind of M-alternating trail (or pair of trails) in G_h. Therefore, by Theorem 5.2.7, we only need to prove the 'If' part for v-loops and double v-rackets. To this end let α be an M-alternating v-loop or an M-alternating double v-racket in G_h. If α is a v-loop, then we are ready by the Shortcut Principle. Therefore, in order to finish the proof, one must show that by unfolding a hidden edge e of the double v-racket $\alpha = (\alpha^1, \alpha^2)$, we will obtain a double v-racket too. In this case, as we have observed in the previous paragraph, the alternating cycles of α do not contain hidden edges, thus we may assume without loss of generality that $e \in E(\alpha_h^1)$. By the Shortcut Principle again, unfolding e results in an M-alternating v-racket β^1, consequently if β^1 is disjoint from α_c^2, then (β^1, α^2) provides the required double v-racket. Otherwise, we can construct a suitable M-alternating double v-racket (β', α^2) in such a way that $\beta'_c = \alpha_c^2$, and

β'_h is the prefix of β^1_h from v to the first vertex common with α^2_c. Hence the proof is complete. $\diamondsuit$

In order to give a more sophisticated description of the graph-structure of the self-transitions, we need a further analysis of v-loops. This analysis will have important algorithmic consequences for solving ACP.

Proposition 5.2.9 *Let v be an external vertex of soliton graph G and $M \in S(G)$. Then G contains an M-alternating v-loop iff there exists an internal edge $e \in E(G)$ such that both endpoints of e are M-accessible from v.*

Proof. The endpoints of any internal edge in an M-alternating v-loop γ is M-accessible from v by an appropriate subpath of γ, thus it is sufficient to prove the 'If' part. For this, let $\alpha = v, e_1, v_1, \ldots, e_n, v_n$ $(n > 0)$ and $\beta = v, f_1, w_1, \ldots, f_m, w_m$ $(m > 0)$ be positive external M-alternating paths from v such that (v_n, w_m) is an internal edge and $|E(\alpha) \cup E(\beta)|$ is minimal among all internal vertices and alternating paths with the above conditions. Now observe that if $i \in [n]$ denotes the maximal index with $w_i = v_j$ for some $j \in [m]$, then $e_j \in M$. Indeed, otherwise $\alpha[v, v_{j-1}]$ and $\beta[v, w_i]$ would contradict the choice of α and β. Therefore an M-alternating v-loop can be collected from the edges of the set $E(\alpha) \cup E(\beta[w_j, w_m]) \cup \{(v_n, w_m)\}$. $\diamondsuit$

Proposition 5.2.10 *Let M be a state of soliton graph G and $v \in Ext(G)$ such that each edge of G is traversed by an external M-alternating trail starting from v. Then G contains a v-loop iff G is non-bipartite.*

Proof. The 'Only if' part is trivial, so for the proof we can assume on the contrary that G is non-bipartite, but it does not contain an M-alternating v-loop. Then let G' denote the maximal subgraph of G with the property P that G' is bipartite, $v \in V(G')$, and each edge of G' is traversed by an $M_{(G')}$-alternating trail starting from v. Notice that any maximal alternating trail α starting from v is either a crossing or a v-racket, thus α has property P, implying that a suitable G' exists. By assumption $G \neq G'$, consequently there exists a maximal external M-alternating trail β from v to some vertex $v' \in V(G)$ traversing an edge not in G'. Then starting from v, let $e = (w, w')$ denote the first edge of β not in $E(G')$ with w being its endpoint belonging to $V(G')$, and let A denote the bipartition class of G' containing w. It is easy to see by the choice of G', that $V(\beta[w', v']) \cap V(G') \neq \emptyset$, consequently, there exists a vertex u at which the first overlap of $\beta[w', v']$ with G' occurs. (See Figure 5.3) Now let γ be an $M_{(G')}$-alternating path from v to u, and consider the following two cases.

Case(1) $u \in A$. In this case, because of the bipartition, the M-alternating path $\beta[v, w]$ is positive iff γ is such; moreover, by the choice of G', the length of $\beta[w, u]$ is even. Therefore, either by $\beta[v, w]$ and $\gamma + \beta^{-1}[u, w']$, or by $\beta[v, w']$ and $\gamma + \beta^{-1}[u, w]$, both endpoints of e are M-accessible from v. Now making use of Proposition 5.2.9, we obtain a contradiction.

Case(2) $u \notin A$. Then, applying the same argument as in *Case*(1), we obtain that $\beta[v, w]$ is positive iff γ is negative, the length of $\beta[w, u]$ is odd; consequently both endpoints of e are M-accessible from v by the appropriate subpaths collected from $\beta[v, u]$ and γ. Therefore, by Proposition 5.2.9, a contradiction is obtained again.

In the preceding two paragraphs, the claim was verified for both possible cases, thus the proof is complete. $\diamondsuit$

Making use of the above observations, we can refine Theorem 5.2.7. To this end we will use the following notation: For any soliton graph G, state M of G and $v \in Ext(G)$, let G^M_v denote the graph determined by the edges traversed by an M-alternating trail starting from v. Since

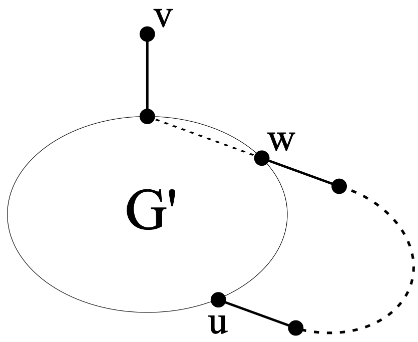

Figure 5.3: The proof of Proposition 5.2.10

for any maximal external M-alternating trail α, $M_{(\alpha)}$ is clearly a perfect internal matching in the graph determined by α, G_v^M is also a soliton graph with $Ext(G_v^M) = Ext(G) \cap V(G_v^M)$ and $M_{(G_v^M)} \in S(G_v^M)$.

Theorem 5.2.11 *For any state M of soliton automaton $\mathcal{A}(G) = ((S(G^+), (X \times X), \delta)$ and for any external vertex $v \in X$ of G, $M \in \delta(M, (v, v))$ iff one of the following conditions holds:*

(a) G_v^M is a non-bipartite graph.

(b) G_v^M is a bipartite graph containing an $M_{(G_v^M)}$-alternating double v-racket.

(c) G_v^M is a bipartite graph not containing an $M_{(G_v^M)}$-alternating cycle.

Proof. Immediate by Theorem 5.2.7 and Proposition 5.2.10. ♢

Now making use of the above theorem we give an efficient method deciding for any state M and external vertex v of soliton graph G if $M \in \delta(M, (v, v))$ holds for the transition function δ of $\mathcal{A}(G)$.

For the above goal, let us fix a soliton graph G for the rest of the section, and let $n = |V(G)|$ and $m = |E(G)|$. By the algorithm of Section 4.7, we can assume that G is viable. Then let $M \in S(G)$ be an arbitrary state and $v \in Ext(G)$. It is clear by Propositions 3.2.4 and 5.2.4 that G_v^M can be constructed in $\mathcal{O}(m)$ time. Furthermore, it can be checked in a straightforward way – e.g. using a depth first search – that G_v^M is bipartite. Finally, applying the method of [51] for testing the existence of an alternating cycle, condition (c) in Theorem 5.2.11 can be also checked. Both of the above algorithms also have $\mathcal{O}(m)$ complexity. Therefore we have:

Proposition 5.2.12 *The graph G_v^M can be constructed in $\mathcal{O}(m)$ time. Furthermore,it can be checked in $\mathcal{O}(m)$ time, if one of the following conditions holds:*

(i) G contains an M-alternating v-loop.

(ii) G does not contain an M-alternating v-racket.

By the above proposition, in order to get an efficient solution for our problem, we only need to design a method testing the existence of an $M_{G_v^M}$-alternating double v-racket. Our algorithm is based on a breadth first search with respect to M-alternating paths starting from v. Applying our observation in the proof of Proposition 5.2.4, we may also suppose that all vertices different from v are internal in G_v^M. Let (A, B) denote the bipartition of G_v^M with A containing those vertices which are M-accessible from v. Construct an M-alternating spanning tree T of G_v^M rooted at v using a breadth-first search and let w denote the deepest vertex (i.e.

a vertex with the smallest depth) having more than one children. Of course, if such a vertex does not exist, then we are ready with a negative answer. It is clear that w is M-accessible, by which $w \in A$.

Algorithm 5.2.13 Let T' be the subtree of T rooted at w, $G' = V[T']$, $w' = w$ and define an index set I with initializing it as $I = \emptyset$. Then repeat the following steps.

1. Let $u_1, \ldots, u_k$ ($k \in N_0$) denote the children of w' in T'. For each $i \in [k]$, let T_i denote the subtree of T' rooted at u_i, and attach an external edge (v_i, u_i) to u_i with $v_i \notin V(T')$.
2. Let $\bar{G}$ denote the graph obtained from $G' - w'$ by attaching all the new external edges in step 1, and let $\bar{M}$ denote the matching $M_{(\bar{G})}$. Moreover, for each $i \in [k]$, use the method of Proposition 5.2.4 to construct the graph $G_i = \bar{G}_{v_i}^{\bar{M}}$, and apply the algorithm of [51] for G_i to decide if G_i contains an $\bar{M}_{(G_i)}$-alternating cycle. If such a cycle is found, then let $I = I \cup \{i\}$.
3. If $I = \emptyset$, then stop the algorithm with the answer that there is no M-alternating double v-racket. If $|I| > 1$, then stop the algorithm with the answer that there exists an M-alternating double v-racket. Otherwise, for the unique index $i \in I$, let w' be the deepest vertex in T_i which has more than one children. If such a vertex does not exist, then stop the algorithm with the answer that there does not exist an M-alternating double v-racket. Otherwise let T' be the subtree of T_i rooted at w', $G' = G[V(T')]$, let $I = \emptyset$, and continue the algorithm with the step 1.

Theorem 5.2.14 *Algorithm 5.2.13 correctly decides if G_v^M contains an $M_{G_v^M}$-alternating double v-racket. The algorithm terminates in $\mathcal{O}(n \cdot m)$ time.*

Proof. It is clear that the algorithm terminates, as in each iteration step it runs on a proper subtree of the tree of the preceding iteration. Moreover, it is also easy to see that if it stops with a positive answer, then G indeed contains an M-alternating double v-racket. In order to prove the correctness for negative answer we will prove by induction on the number of iteration steps that if G contains an M-alternating double v-racket $\beta = (\beta^1, \beta^2)$, then in each iteration step G' contains both cycles of β, the branching vertex x and for $i = 1, 2$ the subpath β_i' of β_h^i connecting x with β_c^i. Observe that, as G_v^M is bipartite, x is necessarily in A.

Consider first the basis step of the induction. Then w' denotes the deepest vertex in G_v^M which is potential to be the branching vertex of β. Now it is trivial that x is contained in T' as a vertex at even distance from w'. As we used breadth-first search, the remaining two statements are also obvious.

In the induction step, make a backtrack in the iteration and use the parameters of the algorithm with respect to this iteration step. Consider step 3. Since the algorithm did not stop at this point, thus $|I| = 1$, and we may assume without loss of generality that $i = 1$. Now observe that no edge of β is contained in a subtree different from T_1. Indeed if such and edge existed for either β^1 or β^2, say β^1, then starting from v the last vertex y of β^1 common with some T_j ($j \neq 1$) is necessarily contained in A because of the bipartition. However, in that case we would obtain that the unique path connecting v_j with y in T_j could be continued on the above subtrail of β^1, by which G_j should contain β_c^1, which is a contradiction. Therefore all the requested parts of β are contained in T_1, thus we are left to show that neither of x nor any vertex of the cycles is an inner vertex of the subpath connecting u_1 with the deepest vertex w' in T_i having more than one children. Again, because of the breadth-first search manner, the depth of x is necessarily larger than that of w', which means that the subpath in question is also a subpath of β_h. The above fact shows that all vertices in the cycles are also descendants of w', by which the induction step is also proved.

Now the correctness of the algorithm is an easy consequence of the claim verified above, whereas the statement for the complexity of the algorithm follows from the easy observation that both the procedure of Proposition 5.2.4 and the method of [51] are applied at most once for any vertex. $\diamondsuit$

Now we can summarize the results concerning ACP.

Theorem 5.2.15 *For any states M_1, M_2 of G^+ and any external vertices $v, w \in Ext(G)$ it can be decided in $\mathcal{O}(n \cdot m)$ time if $M_2 \in \delta(M_1, (v, w))$ holds for the transition function δ of $\mathcal{A}(G)$.*

Proof. If $M_1 \neq M_2$ or $v = w$, then the proof is immediate by Theorems 5.2.11 and 5.2.14, Proposition 5.2.12 and Corollary 5.2.5. Otherwise, i.e. $M_1 = M_2$ with $v \neq w$, we need to test if an M_1-alternating crossing exists between v_1 and v_2, which can be solved in $\mathcal{O}(m)$ time by Proposition 5.2.4. $\diamondsuit$

Theorem 5.2.16 *Let G be a soliton graph and $k = |S(G^+)|$. Then ACP can be solved in $\mathcal{O}(k^2 \cdot n \cdot m)$ time.*

Proof. Immediate by Theorems 5.2.1 and 5.2.15. $\diamondsuit$

Corollary 5.2.17 *For any soliton graph G, ACP can be solved in polynomial time iff G^+ has a polynomial number of states.*

Proof. Immediate by Theorem 5.2.16. $\diamondsuit$

5.3 Soliton automata with a single external vertex

As a first step towards the characterization of nondeterministic soliton automata with a single external vertex, the transition between distinct states is described below as a straightforward consequence of Theorem 5.2.3.

Theorem 5.3.1 *If G is a soliton graph with a single external vertex v, then $M_2 \in \delta(M_1, (v, v))$ holds for any distinct states M_1, M_2 of $\mathcal{A}(G) = (S(G^+), X \times X, \delta)$.*

Proof. Immediate by Theorem 5.2.3 and by the observation that the mediator alternating network between M_1 and M_2 consists of alternating cycles accessible from v in M_1. $\diamondsuit$

For the analysis of self-transitions, according to Theorem 5.2.7, we need to investigate v-loops and (double) v-rackets. In reaching the above goal the following concept will play a central role.

Definition 5.3.2 States M_1 and M_2 of soliton graph G are called *compatible*, if M_1 and M_2 cover the same external vertices.

Proposition 5.3.3 *Let M and M' be compatible states of soliton graph G and let α be an M-alternating crossing between external vertices v and w. Then there exists an M'-alternating crossing α' connecting v and w.*

Proof. Let $\beta_1, \ldots, \beta_k$ $(k \geq 0)$ be the alternating cycles constituting the mediator alternating network between M and M'; and construct the graph $G' = \alpha + \beta_1 + \ldots + \beta_k$. Then it is clear that $Ext(G') = \{v, w\}$, $M'_{(G')} \in S(G')$, and the external edges incident with v and w are non-constant in G'. Therefore, making use of Theorem 2.3.2, we easily obtain that v and w are connected by an $M'_{(G')}$-alternating crossing, as required. $\diamondsuit$

Corollary 5.3.4 *Let M and M' be compatible states of a soliton graph G. Then for any external vertex $v \in Ext(G)$, $G_v^M = G_v^{M'}$ holds.*

Proof. By symmetry, it is enough to prove the claim in one direction. To this end let e be an edge of G traversed by an M-alternating trail α starting from v. If e is external, then the statement follows directly from Proposition 5.3.3. Suppose now that e is internal. In that case one endpoint of e, let it be denoted by w, is M-accessible from v either by α or by the appropriate prefix of α. Now let extend G by a new external edge (w, u) such that $u \notin V(G)$. By the above observation there is an M-alternating crossing between v and u in $G+(w, u)$. Now applying Proposition 5.3.3, we obtain that v and u is connected by an M'-alternating crossing β. Since M and M' are compatible, we conclude that $\beta[v, w]$ is a positive M'-alternating alternating path in G. Therefore either β or $\beta + e$ will provide an M'-alternating trail starting from v and traversing e; as required. $\diamondsuit$

Corollary 5.3.5 *Let M_1 and M_2 be compatible states of soliton graph G and $v \in Ext(G)$. Then G contains an M_1-alternating v-loop iff it contains an M_2-alternating v-loop.*

Proof. Immediate by Proposition 5.2.10 and Corollary 5.3.4. $\diamondsuit$

Proposition 5.3.6 *Let G be a bipartite soliton graph, M be a state of G and $v \in Ext(G)$. Then G contains an M-alternating double v-racket iff there exists an M'-alternating double v-racket for all states M' compatible with M.*

Proof. Let $\beta = (\beta^1, \beta^2)$ be an M-alternating double v-racket with branching vertex w, α be an M-alternating cycle, and $M' = S(M, \alpha)$. Since any mediator alternating network Γ between compatible states consists of alternating cycles only, if we prove that an M'-alternating double v-racket also exists, then we are ready by Theorem 2.3.1 and by a straightforward induction argument on $|\Gamma|$. For this goal consider first the case that $V(\alpha) \cap V(\beta_h) = \emptyset$.

If $V(\alpha) \cap V(\beta) = \emptyset$, then our statement is trivial. Otherwise, for $k = 1, 2$, let β'_k denote the suffix of β^k from w to its internal endpoint, and if $V(\beta^k) \cap V(\alpha) \neq \emptyset$, then let β''_k denote the prefix of β^k from v to the first vertex common with α. (See an example in Figure 5.4.) Then an M'-alternating double v-racket $\gamma = (\gamma^1, \gamma^2)$ can be constructed in the following way: If $V(\alpha) \cap V(\beta'_k) \neq \emptyset$ for $k = 1, 2$, then let $\gamma_h^1 = \beta''_1$, $\gamma_h^2 = \beta''_2$, and $\gamma_c^1 = \gamma_c^2 = \alpha$. Otherwise, i.e. $V(\alpha) \cap V(\beta'_k) \neq \emptyset$ and $V(\alpha) \cap V(\beta'_{3-k}) = \emptyset$ for some $k \in \{1, 2\}$, let $\gamma_h^k = \beta''_k$, $\gamma_c^k = \alpha$, and $\gamma^{3-k} = \beta^{3-k}$.

By the preceding paragraph we can assume for the rest of the proof that $V(\alpha) \cap V(\beta_h) \neq \emptyset$. In this case, starting from v, let u denote the first vertex at which β_h overlaps with α, and let u' be the vertex of $V(\alpha) \cap V(\beta)$ such that the positive M-alternating path $\beta' = \alpha[u, u']$ is maximal as a subpath in β. (Observe that $\beta_h[v, u]$ is negative at the u end.) Moreover, let α' be the negative M-alternating subpath of α connecting u and u', i.e. $E(\alpha') = E(\alpha) \setminus E(\beta')$. See Figure 5.5 for an example.

From now on, assume that β is such that the subpath β' constructed above is maximal. Then the following holds.

Claim A *α' is edge-disjoint from β.*

In order to prove the above claim, let us assume by contradiction, that starting from u', the next vertex u'' of α' having the property that $u'' \in V(\beta)$ and u'' is different from u.(See Figure 5.5 again) It is clear that $\alpha'[u', u'']$ is a negative alternating path, so that u' and u'' belong to distinct bipartition class of G. Furthermore, we may suppose without loss of generality that $u' \in V(\beta^1)$. We know by the choice of β' that $\beta^1[v, u']$ is positive at its u' end, but it is also easy to observe that if $u'' \in V(\beta^k)$ $(k \in \{1, 2\})$, then $\beta^k[v, u'']$ is negative at its u'' end.

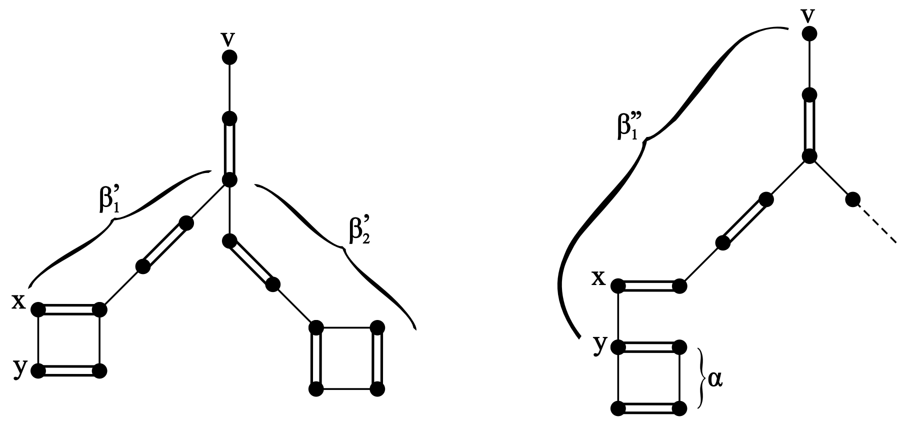

Figure 5.4: The case $V(\alpha) \cap V(\beta_h) = \emptyset$ in the proof of 5.3.6

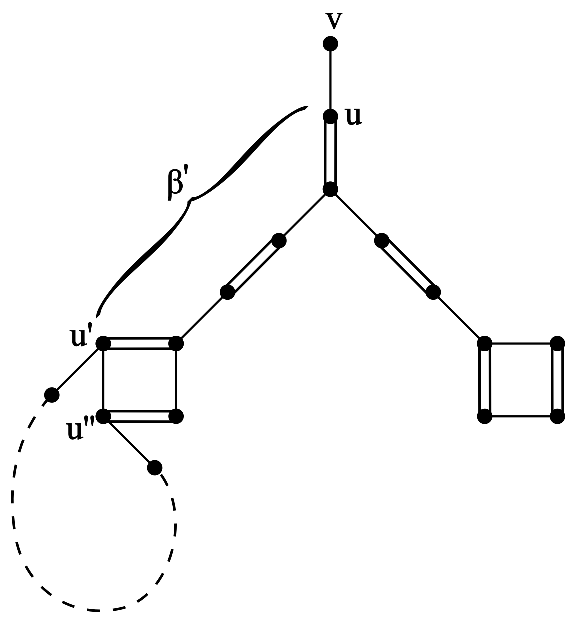

Figure 5.5: The case $V(\alpha) \cap V(\beta_h) \neq \emptyset$ in the proof of 5.3.6

Indeed, if $\beta^k[v, u'']$ terminated in a positive edge at u'', then according to Proposition 5.2.9, the bipartite graph $G + (u', u'')$ would contain an M-alternating v-loop, which is a contradiction. For $k = 1, 2$, now let v_k denote the internal endpoint of β_h^k, and let v'' be the vertex adjacent to u'' by positive edge in M. We will show that an M-alternating double v-racket $\gamma = (\gamma^1, \gamma^2)$ can be constructed such that the positive M-alternating path $\alpha[u, v'']$ is a subpath of γ^1. For this, we distinguish four cases.

Case A/1: $u'' \in V(\beta_h^k)$ $(k \in \{1, 2\})$, and either $u' \in V(\beta_h^1)$ or $\beta_c^1 \neq \beta_c^2$.

Note that, in this case, if $\beta_c^1 \neq \beta_c^2$, then $k = 2$. Now it is easy to check that the trails defined below constitute an M-alternating double v-racket γ such that $\alpha[u, v'']$ is a positive M-alternating subpath of γ^1, as required.

$\gamma_h^1 = \beta^1[v, u'] + \alpha'[u', u''] + \beta_h^k[u'', v_k]$
$\gamma_h^2 = \beta_h^k$
$\gamma_c^1 = \gamma_c^2 = \beta_c^k$.

Case A/2: $u'' \in V(\beta_c^k)$ $(k \in \{1, 2\})$ such that either $u' \in V(\beta_h^1)$, or $\beta_c^1 \neq \beta_c^2$ with $k = 2$.

Then let
$\gamma_h^1 = \beta^1[v, u'] + \alpha'[u', u'']$,
$\gamma_h^2 = \beta_h^k$,
and $\gamma_c^1 = \gamma_c^2 = \beta_c^k$.

Again, considering all possible alternatives of this case, we obtain that $\gamma = (\gamma^1, \gamma^2)$ is a double v-racket with the required properties.

Case A/3: $u', u'' \in V(\beta_c^1)$.

Now a suitable $\gamma = (\gamma^1, \gamma^2)$ is defined as follows.
$\gamma_h^1 = \beta_h^1$,
$\gamma_h^2 = \beta_h^2$,
$\gamma_c^1 = \beta'[v_1, u'] + \alpha'[u', u''] + \beta^1[u'', v_1]$.
and
$\gamma_c^2 = \gamma_c^1$, if $\beta_c^1 = \beta_c^2$
$\gamma_c^2 = \beta_c^2$, if $\beta_c^1 \neq \beta_c^2$.

Case A/4: $u' \in V(\beta_c^1)$, $u'' \in V(\beta_h^2)$, and $\beta_c^1 = \beta_c^2$.

Now the construction of $\gamma = (\gamma^1, \gamma^2)$ below is represented in Figure 5.6.
$\gamma_h^1 = \beta_h^1$,
$\gamma_h^2 = \beta_h^2[v, u'']$,
$\gamma_c^1 = \gamma_c^2 = \beta'[v_1, u'] + \alpha'[u', u''] + \beta^2[u'', v_1]$.

It is clear that *Cases* $A/1 - A/4$ cover all the possible alternatives of the locations of u' and u''. Nevertheless, we obtained in all cases that $\alpha[u', v'']$ is a positive M-alternating subpath of γ^1. However, the length of $\alpha[u', v'']$ is greater than that of β', which contradicts the choice of β. Therefore Claim A is proved.

By the above claim we can suppose for the rest of the proof that α' is edge-disjoint from β'. As earlier, assume that $u' \in V(\beta^1)$ and for $i \in \{1, 2\}$, let v_i denote the internal endpoint of β_h^i. We will construct an M'-alternating double v-racket $\delta = (\delta^1, \delta^2)$ in the subgraph determined by β and α'. For this, we must deal with three cases and several subcases.

Case 1: $v_1 = v_2$.

In this case $\beta_c^1 = \beta_c^2$, for which we use the notation β_c. Now starting from v, let y denote the last vertex of β_h^1 such that y is incident with an edge in $E(\beta_h^2) \setminus E(\beta_h^1)$, and let x denote the last vertex of β_h^1 preceding y with $x \in V(\beta_h^2)$. (See Figure 5.7.) Below we give the construction of $\delta = (\delta^1, \delta^2)$, for which, based on the location of u', we distinguish four subcases.

Subcase 1a: $u' \in V(\beta_h^1[v, x])$.

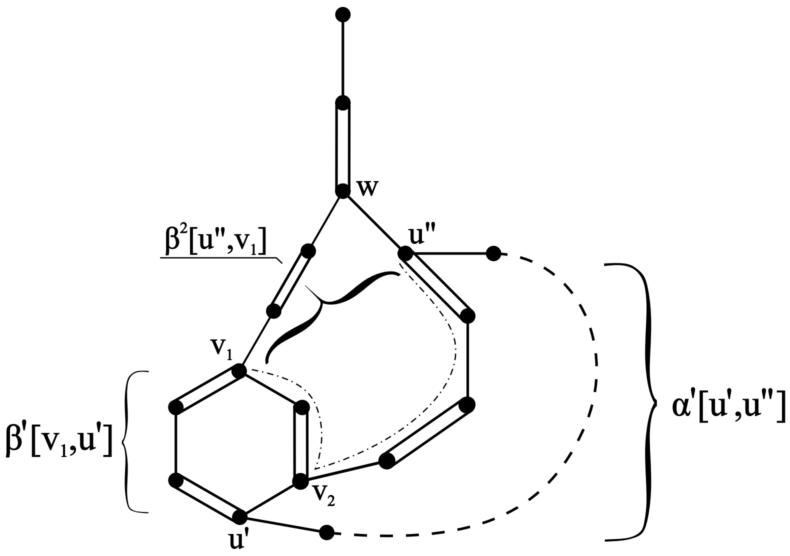

Figure 5.6: Case $A/4$ in the proof of 5.3.6.

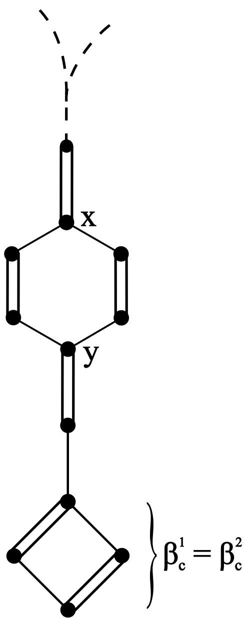

Figure 5.7: Case 1 in the proof of 5.3.6

Then
$\delta_h^1 = \beta_h[v,u] + \alpha' + \beta_h^1[u',v_1]$,
$\delta_h^2 = \beta_h[v,u] + \alpha' + \beta_h^1[u',x] + \beta_h^2[x,v_2]$
$\delta_c^1 = \delta_c^2 = \beta_c$.

Subcase 1b: $u' \in V(\beta_h^1[x,y])$.
Then
$\delta_h^1 = \beta_h[v,u] + \alpha' + \beta_h^1[u',v_1]$,
$\delta_h^2 = \beta_h[v,u] + \alpha' + (\beta_h^1)^{-1}[u',x] + \beta_h^2[x,v_2]$
$\delta_c^1 = \delta_c^2 = \beta_c$.

Subcase 1c: $u' \in V(\beta_h^1[y,v_1])$.
Then
$\delta_h^1 = \beta_h[v,u] + \alpha' + \beta_h^1[u',v_1]$,
$\delta_h^2 = \beta_h[v,u] + \alpha' + (\beta_h^1)^{-1}[u',y]$
$\delta_c^1 = \beta_c$,
$\delta_c^2 = \beta_h^1[x,y] + (\beta_h^2)^{-1}[y,x]$.

Subcase 1d: $u' \in V(\beta_c^1)$.
In this case, let β'' denote the negative M-alternating subpath from u' to v_1 in β_c determined by the edges not contained in β'.
Then
$\delta_h^1 = \beta_h[v,u] + \alpha' + (\beta')^{-1}[u',y]$,
$\delta_h^2 = \beta_h[v,u] + \alpha' + \beta'' + (\beta_h^1)^{-1}[v_1,y]$
$\delta_c^1 = \delta_c^2 = \beta_h^1[x,y] + (\beta_h^2)^{-1}[y,x]$.

Now it is easy to check that $\delta = (\delta^1, \delta^2)$ is indeed a suitable M'-alternating double v-racket in all of the above cases.

Case 2: $v_1 \neq v_2$ and $\beta_c^1 = \beta_c^2$.
In this case consider the bipartite graph G' determined by α' and β. Applying the method of contracting the redexes of G' in an iterative way, construct the graph G_1 having no redexes. Then let M_1 (M_1') denote the restriction of M (respectively, M') to the edges of G_1. It is easy to see that β is contracted to an M_1-alternating double v-racket $\gamma = (\gamma^1, \gamma^2)$ such that γ_h^1 and γ_h^2 have the same internal endpoint. Therefore, according to Case 1, an M_1'-alternating double v-racket also exists, which obviously becomes an M'-alternating double v-racket after unfolding.

Case 3: $\beta_c^1 \neq \beta_c^2$.
To handle this case, we need to further break it down into two subcases.

Subcase 3a: $u' \in V(\beta_1^h)$.
Consider the subgraph G_h determined by β_h^1 and β_h^2. Then v_1 and v_2 are obviously accessible in G_h, consequently there exists an $M'_{(G_h)}$-alternating path α_1 (α_2) from v to v_1 (respectively, v_2). Therefore the external alternating trails $\delta^1 = \alpha_1 + \beta_c^1$ and $\delta^2 = \alpha_2 + \beta_c^2$ form an M'-alternating double v-racket.

Subcase 3b: $u' \in V(\beta_1^c)$.
Now starting from v, let x denote the last vertex of β_h^1 which is also on β_h^2, and let β'' denote the negative M-alternating subpath from u' to v_1 in β_c^1 determined by the edges not contained in β'. Then a suitable M'-alternating double v-racket$\delta = (\delta^1, \delta^2)$ is defined as follows.
$\delta_h^1 = \beta_h[v,u] + \alpha' + (\beta')^{-1}[u',x] + \beta_h^2[x,v_2]$,

$\delta_h^2 = \beta_h[v,u] + \alpha' + \beta'' + (\beta_h^1)^{-1}[v_1,x] + \beta_h^2[x,v_2]$
$\delta_c^1 = \delta_c^2 = \beta_c^2.$

Now the proof is complete, as all the possible cases have been covered. $\diamondsuit$

By the above result it is meaningful to say for a bipartite soliton graph G with a single external vertex v that "G contains a double v-racket" without specifying any state.

Now we are ready to prove our main result concerning self-transitions.

Theorem 5.3.7 *Let M, M' be compatible states of soliton automaton $\mathcal{A}(G) = (S(G^+), X \times X, \delta)$ and $v \in X$ be an external vertex of G. Then $M \in \delta(M,(v,v))$ iff $M' \in \delta(M',(v,v))$.*

Proof. Let G_v denote the subgraph $G_v^M = G_v^{M'}$ (see Corollary 5.3.4) and apply Theorem 5.2.11 for G_v. It is clear that if condition (c) holds with respect to M or M', then $M = M'$, and there is nothing to prove. The statement is also obvious if G_v is non-bipartite, while in other case the theorem follows from Proposition 5.3.6. $\diamondsuit$

Making use of the results of this section, we obtain a characterization of soliton automata with a single external vertex.

Definition 5.3.8 Let $\mathcal{A} = (S, X, \delta)$ be an automaton such that its alphabet is a singleton, i.e. $X = \{x\}$. We say that $\mathcal{A}$ is a *full* (*semi-full*) automaton if for each $s \in S$, $\delta(s,x) = S$ (respectively, $\delta(s,x) = S \setminus \{s\}$ with $|S| > 1$).

Theorem 5.3.9 *Let G be a soliton graph with a single external vertex v. Then $\mathcal{A}(G)$ is either a full or a semi-full automaton. Moreover, $\mathcal{A}(G)$ is semi-full iff G^+ is a bipartite graph without double v-rackets.*

Proof. Since any maximal alternating trail in G^+ is necessarily either a v-loop or a v-racket, G^+ is either non-bipartite or it contains alternating cycle with respect to any of its states. Therefore the argument is straightforward by using Theorems 5.3.1 and 5.2.7 with Propositions 5.2.10 and 5.3.6. $\diamondsuit$

5.4 Elementary decomposition of soliton automata

In this section we will show that the class of soliton automata is equivalent to certain two-level ε-products of *elementary soliton automata*, i.e. soliton automata associated with an elementary graph. For the above goal, because of technical reasons, we introduce a special type of isomorphism, called soliton isomorphism, which is based on an extended approach for the definition of soliton walks and soliton automata developed below.

Concerning the logical aspects of our model it is also a realistic alternative to allow the use of empty soliton walks (see e.g. [9], [58]). In this case the effect of an empty soliton walk α for a state M is: $S(M,\alpha) = M$. By the above facts, we can define the extension of any soliton automaton as follows.

Definition 5.4.1 Let $\mathcal{A}(G) = (S(G^+), X \times X, \delta)$ be a soliton automaton. The *extension* of $\mathcal{A}(G)$ is the automaton $\mathcal{A}_e(G) = (S(G^+), X \times X, \delta_e)$, where for any state M of G and any pair of external vertices $(v,w) \in X \times X$,

$$\delta_e(M,(v,w)) = \begin{cases} \delta(M,(v,w)), & \text{if } v \neq w \\ \delta(M,(v,w)) \cup \{M\}, & \text{otherwise} \end{cases}$$

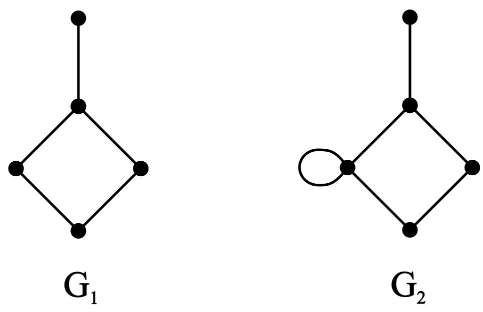

Figure 5.8: Example for soliton automata which are not isomorphic, but equivalent by soliton isomorphism.

Definition 5.4.2 Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be finite automata. We say that a *soliton isomorphism* exists between $\mathcal{A}_1$ and $\mathcal{A}_2$, if for $i = 1, 2$ there is a soliton automaton $\mathcal{A}(G_i)$ such that $\mathcal{A}_i$ is strongly isomorphic with $\mathcal{A}(G_i)$, and $\mathcal{A}_e(G_1)$ is strongly isomophic with $\mathcal{A}_e(G_2)$. The existence of a soliton isomorphism between automata $\mathcal{A}_1$ and $\mathcal{A}_2$ is expressed by $\mathcal{A}_1 \cong_s \mathcal{A}_2$.

The fact that automata classes $\mathcal{K}$ and $\mathcal{F}$ *coincide up to soliton isomorphism* is defined in an analogous fashion to isomorphism.

It is clear by definition that $\mathcal{A}(G_1) \cong_s \mathcal{A}(G_2)$ holds for any strongly isomorphic soliton automata $\mathcal{A}(G_1)$ and $\mathcal{A}(G_2)$. However, in deterministic case the converse is also true, as the following simple observation shows.

Proposition 5.4.3 *Let* $\mathcal{A}(G_1) = (S(G_1^+), X_1 \times X_1, \delta^1)$ *and* $\mathcal{A}(G_2) = (S(G_2^+), X_2 \times X_2, \delta^2)$ *be deterministic soliton automata. Then* $\mathcal{A}(G_1) \cong_s \mathcal{A}(G_2)$ *iff* $\mathcal{A}(G_1)$ *and* $\mathcal{A}(G_2)$ *are strongly isomorphic.*

Proof. Clearly, it is enough to prove the 'Only if' part. To this end let $\psi = (\psi_S, \psi_X)$ be a strong isomorphism between $\mathcal{A}_e(G_1)$ and $\mathcal{A}_e(G_2)$ by bijections $\psi_S : S(G_1^+) \to S(G_2^+)$ and $\psi_X : X_1 \to X_2$. Now let $M_1 \in S(G_1^+)$ and $v_1, v_1' \in X_1$ be arbitrary and consider the unique element M_1' of $\delta^1(M_1, (v_1, v_1'))$. We will prove that $\delta^2(\psi_S(M_1), (\psi_X(v_1), \psi_X(v_1'))) = \{\psi_S(M_1')\}$. If $v_1 \neq v_1'$ or $M_1' = M_1$, then our claim is obvious by the given strong isomorphism between $\mathcal{A}_e(G_1)$ and $\mathcal{A}_e(G_2)$. Therefore consider the case with $v_1 = v_1'$ and $M_1 \neq M_1'$. Then $\delta_e^1(M_1, (v_1, v_1)) = \{M_1, M_1'\}$ and $\delta_e^2(\psi_S(M_1), (\psi_X(v_1), \psi_X(v_1))) = \{\psi_S(M_1), \psi_S(M_1')\}$ hold because of the soliton isomorphism. Since G_2 is also deterministic, the above facts imply that $\delta^2(\psi_S(M_1), (\psi_X(v_1), \psi_X(v_1))) = \{\psi_S(M_1')\}$, by which the proof is complete. $\diamondsuit$

Figure 5.8 presents a simple example demonstrating that Proposition 5.4.3 does not hold in nondeterministic case.

Note that from practical point of view, soliton isomorphism is meaningful only if we can describe the self-transitions of the considered automata. Therefore, related to the results where soliton isomorphism applied, we will give the above characterization.(see Section 5.5)

Making use of the above concept we can characterize *essentially elementary soliton automata*, by which we mean soliton automata associated with an internal self-contained graph consisting of one internal elementary component and an external edge. Note that for automata with alphabet consisting of a single symbol x, the concepts of isomorphism and strong isomorphism coincide, as x can be identified with (x, x).

Theorem 5.4.4 *Let $\mathcal{F}$ denote the class of full and semi-full automata, and $\mathcal{H}$ denote the class of essentially elementary soliton automata associated with a non-bipartite graph. Furthermore, class $\mathcal{J}$ is defined by the following way: $\mathcal{J} = \{ \mathcal{A}(G) | \ \mathcal{A}(G)$ is an essentially elementary soliton automaton with external vertex v such that G is a bipartite graph without double v-rackets$\}$. Then the following statements hold.*

(i) The class of full automata coincides with $\mathcal{H}$ up to isomorphism.

(ii) The class of essentially elementary soliton automata coincides with $\mathcal{F}$ up to isomorphism.

(iii) The class of semi-full automata coincides with $\mathcal{J}$ up to isomorphism.

(iv) The class of full automata coincides with the class of essentially elementary soliton automata up to soliton isomorphism.

Proof. By Theorem 5.3.9, in each case $((i) - (iv))$ it is clearly sufficient to prove that for any $n \in N$ there exists an essentially elementary soliton automaton with n states having the required properties. For $n = 1$ the graph consisting of a mandatory internal elementary component C and an external edge incident with C determines a suitable automaton. Now let $n > 1$ arbitrary and construct a closed bipartite elementary graph G_n with n states in the following way.

Consider an even cycle β, two adjacent vertices $v, w \in V(\beta)$ and define G_n such that it has a representation in the form $G_n = \beta + \alpha_1 + \ldots + \alpha_{n-2}$, where

(a) α_i, $i \in [n-2]$ is an odd path with endpoints v and w

(b) $V(\alpha_i) \cap V(\beta) = \{v, w\}$, $i \in [n-2]$

(c) $V(\alpha_i) \cap V(\alpha_j) = \{v, w\}$, $i, j \in [n-2]$

Observe that for any edge e being incident with v, there is a unique state M of G_n such that $e \in M$. Thus, it is easy to see that each edge of G_n is allowed and G_n has n states, as expected.

Moreover, it is easy to see that by attaching an external edge (v, w) to any vertex w of G_n, we obtain a graph $G'_n = G_n + (v, w)$ which does not contain double v-rackets. Finally, let G^u_n denote the graph obtained from G'_n by adding a loop around u, where u is a vertex with $w \not\sim u$. Clearly, u is accessible from v. Now $\mathcal{A}(G^u_n)$ is a suitable choice for (i),(ii) and (iv), while $\mathcal{A}(G'_n)$ provides an appropriate automaton for (ii) and (iii). $\diamondsuit$

The main result of this section will describe the decomposition of soliton automata into elementary ones. For this goal first we prove the following simple observation on canonical classes, which will play an important role in describing the appropriate automata products.

Claim 5.4.5 *Let P be a canonical class of some elementary component C in soliton graph G. A vertex w of P is accessible from external vertex v in state $M \in S(G)$ iff all vertices of P are M-accessible from v .*

Proof. Let us assume that α is a positive external M-alternating path from v to w, and let u be an arbitrary vertex of P different from w. We claim that there exists an internal M-alternating path β between u and some vertex of α such that β is positive at the u end.

If C is external, then, according to Claim 4.2.1, there exists a positive external $M_{(C)}$-alternating path γ with endpoint u. Observe that $E(\alpha) \cap E(\gamma) \neq \emptyset$, because otherwise (α, γ) would form an alternating fork leading to the contradiction $u \not\sim w$. Therefore an appropriate subpath of γ can be chosen for β.

Now assume that C is internal. Then let w' denote the vertex incident with w by the edge covered by M. Clearly, $u \not\sim_C w'$, thus there exists a positive internal $M_{(C)}$-alternating path between u and w' providing a suitable choice for β again.

Now starting from u let u_α denote the first vertex along β for which $u_\alpha \in V(\alpha)$. $\alpha_1 = \alpha^{-1}[w, u_\alpha] + \beta[u_\alpha, u]$ cannot form a positive internal alternating path, as it would contradict the fact that $u \sim w$. Therefore $\alpha[v, u_\alpha] + \beta[u_\alpha, u]$ provides a positive external M-alternating path, as required. $\diamondsuit$

By Claim 5.4.5, it is justified to say that a *canonical class is accessible* from an external vertex in a given state. Applying this new concept, for any soliton graph G, $v \in Ext(G)$ and $M \in S(G)$, let $\mathcal{P}_G(M, v)$ denote the set of canonical classes of G which are M-accessible from v.

Note that we can assume without loss of generality that all constant external edges of a soliton graph G are mandatory. Indeed, attaching an extra mandatory edge to each forbidden external edge of G results in a graph G' for which $\mathcal{A}(G)$ and $\mathcal{A}(G')$ are strongly isomorphic. In the rest of this chapter, we shall use the above assumption – unless otherwise specified – without any further reference.

In order to state the main result of this section, we introduce some technical notations and prove two lemmas. For any elementary component C, let $\mathcal{F}_C$ denote the family containing C.

Definition 5.4.6 Let G be a soliton graph, and let $P \in \mathcal{P}(G)$ be a canonical class in some external elementary component of G. Then the set $\rho_P(G)$ is the smallest set of elementary components such that:

(i) if C' is an elementary component of G and (v, w) is an edge for which $v \in P$ and $w \in V(C')$, then $C' \in \rho_P(G)$.

(ii) if C_1, C_2 are viable elementary components such that $\mathcal{F}_{C_1} \overset{*}{\mapsto} \mathcal{F}_{C_2}$, $C_1 \in \rho_P(G)$ and there is an edge between C_1 and C_2, then $C_2 \in \rho_P(G)$.

Note that (ii) may also hold if C_1 and C_2 are in the same family, as $\overset{*}{\mapsto}$ is reflexive. It is also clear by Theorem 4.5.22 that an external family different from the one containing P will not contain any elementary component of $\rho_P(G)$. Moreover, Proposition 4.5.18 implies that for any elementary component C of an external family, there is a unique class P with $C \in \rho_P(G)$. Now summarizing the above facts, the following proposition can be easily proved by using a structural induction based on the recursion in Definition 5.4.6.

Proposition 5.4.7 *Let P be a canonical class in some external elementary component C of soliton graph G, and let $\mathcal{F}$ be a viable family of G. Then the followings hold for $\rho_P(\mathcal{F}) = \rho_P(G) \cap \mathcal{F}$:*

(a) If $\mathcal{F}$ is external with $\mathcal{F} \neq \mathcal{F}_C$, then $\rho_P(\mathcal{F}) = \emptyset$.

(b) If $\mathcal{F} = \mathcal{F}_C$, then $\rho_P(\mathcal{F})$ is equal to the P-subfamily of $\mathcal{F}$.

(c) If $\mathcal{F}$ is internal, then either $\rho_P(\mathcal{F}) = \emptyset$ or $\rho_P(\mathcal{F}) = \mathcal{F}$.

For any internal elementary component C' of graph G, let us introduce the following notation:

$$\mathcal{R}_G(C') = \{P \mid P \text{ is a canonical class of some external elementary component and } C' \in \rho_P(G)\}$$

Lemma 5.4.8 *Let C be an external and C' be a viable internal elementary component of soliton graph G. Furthermore let P' be a non-principal canonical class in C' and v be an external vertex of C. Then P' is accessible from v in state $M \in S(G)$ iff $\mathcal{R}_G(C') \cap \mathcal{P}_{C_h}(M_{(C)}, v) \neq \emptyset$.*

Proof. Throughout the proof, for any alternating path α starting from an external vertex of C, w_α will denote the last vertex of α for which $w_\alpha \in V(C)$.

'Only if' Let α be a positive external M-alternating path starting from v and terminating at vertex $w \in P'$. Moreover, let P_α denote the canonical class containing w_α. Then, substituting the C-loops in α for the appropriate hidden edges, we obtain that $P_\alpha \in \mathcal{P}_{C_h}(M_{(C)}, v)$. Now using Corollary 4.5.23, it is easy to see that $C_s \in \rho_{P_\alpha}(G)$ for each internal elementary component C_s reached by $\alpha[w_\alpha, w]$. Hence $P_\alpha \in \mathcal{R}_G(C') \cap \mathcal{P}_{C_h}(M_{(C)}, v)$, as expected.

'If' Suppose that $C' \in \rho_P(G)$ for some canonical class $P \in \mathcal{P}_{C_h}(M_{(C)}, v)$. Then based on the definition of $\rho_P(G)$ there exist families $\mathcal{F}_1, \ldots, \mathcal{F}_m$ $(m > 0)$ containing members of $\rho_P(G)$ such that $\mathcal{F}_1 = \mathcal{F}_C$, $\mathcal{F}_m = \mathcal{F}_{C'}$ and for each $s \in [m-1]$ $\mathcal{F}_s \mapsto \mathcal{F}_{s+1}$ with some edges connecting elements of $\rho_P(G) \cap \mathcal{F}_s$ and $\rho_P(G) \cap \mathcal{F}_{s+1}$. The proof will apply an induction on m.

For the basis step – i.e. $\mathcal{F}_C = \mathcal{F}_{C'} = \mathcal{F}_1$ – by Proposition 4.5.19, we can consider a positive external M-alternating path α terminating at $w \in P'$. We claim that $w_\alpha \in P$ holds. Suppose not, i.e. w_α belongs to a canonical class P_α different from P. Since, by Corollary 4.5.23, α stays within $\mathcal{F}_1$, α necessarily goes through an edge (u, u_α) such that u (u_α) is a vertex in elementary component C^u (respectively, C^u_α) belonging to $\rho_P(G) \cap \mathcal{F}_1$ (respectively, $\rho_{P_\alpha}(G) \cap \mathcal{F}_1$). Now applying Proposition 5.4.7, part (b), we obtain that (u, u_α) connects two elementary components contained in distinct subfamilies. However, the above fact is clearly a contradiction in Proposition 4.5.18, part (b).

We can conclude by the preceding paragraph that $w_\alpha \in P$. Therefore, choosing a positive M_C-alternating path α_1 in C_h between v and w_α and applying the Shortcut Principle for $\alpha_1 + \alpha[w_\alpha, w]$, we obtain a suitable alternating path.

Now assume for the induction step that $m > 1$. Then there exist elementary components $C_{m-1} \in \mathcal{F}_{m-1} \cap \rho_P(G)$ and $C_m \in \mathcal{F}_m \cap \rho_P(G)$ such that they are connected by an edge (v_{m-1}, v_m) with $v_{m-1} \in V(C_{m-1})$ and $v_m \in V(C_m)$. By Proposition 4.5.20 it is clear that $C_m = r(\mathcal{F}_m)$ with v_m being principal, whereas v_{m-1} is non-principal. Therefore, according to the induction hypothesis, there exists a positive external M-alternating path β between v and v_{m-1} such that $w_\beta \in P$. Furthermore, it follows from Proposition 4.6.9 that v_m is connected to any vertex $w \in P'$ by an internal positive M-alternating path γ running entirely inside $\mathcal{F}_m$. Since, by Corollary 4.5.23, β does not go through $\mathcal{F}_m$, $\beta + (v_{m-1}, v_m) + \gamma$ is an external M-alternating path with the required properties. $\diamondsuit$

We will use later the following strengthening of the 'If' part of Lemma 5.4.8, which was actually also proved above.

Corollary 5.4.9 *Let C be an external and C' be a viable internal elementary component of soliton graph G. Furthermore, let v be an external vertex of C, P be a canonical class in C and $M \in S(G)$ such that $P \in \mathcal{R}_G(C') \cap \mathcal{P}_{C_h}(M_{(C)}, v)$. Then there exists in G a positive external M-alternating path α from v to any non-principal vertex of C' such that along α the last vertex of C belongs to P.*

For constant external edges the above results have a simpler form, which is stated below.

Corollary 5.4.10 *Let G be a soliton graph, $v \in Ext(G)$ be an external vertex incident with a constant edge, and let C denote the elementary component containing v. Then the following three statements are equivalent for any internal vertex w.*

(a) w is accessible from v in some state of G.

(b) w is accessible from v in all states of G.

(c) w is a non-principal vertex of a viable elementary component C' such that $\mathcal{F}_C \overset{*}{\mapsto} \mathcal{F}_{C'}$.

Proof. Combining Proposition 5.4.7 and Lemma 5.4.8 with the recursion of Definition 5.4.6, one can prove by an inductive argument that a non-principal vertex of a viable elementary component C' is accessible from v in any state of G iff $\mathcal{F}_C \overset{*}{\mapsto} \mathcal{F}_{C'}$. Moreover, it is clear by Proposition 4.5.19 and Corollary 4.5.25, that vertices in impervious elementary components and principal vertices are not accessible. Based on the above facts, the proof is now straightforward. $\diamondsuit$

Now we are ready to describe the elementary decomposition of soliton automata, for which we only need to give the definition of the suitable automata product. For this, recall that $\mathcal{P}(G)$ denotes the canonical partition of G.

Definition 5.4.11 Let $\mathcal{Q} = \{\mathcal{A}(G_1), \ldots, \mathcal{A}(G_n)\}(n \in N)$ be a system of soliton automata, and $\mathcal{L} = \{\mathcal{B}_1, \ldots, \mathcal{B}_m\}(m \in N_0)$ be a system of (not necessarily soliton) automata. Furthermore let $\mathcal{P}(\mathcal{Q}) = \mathcal{P}(G_1) \sqcup \ldots \sqcup \mathcal{P}(G_n)$. Then a *canonical dependency* from $\mathcal{Q}$ to $\mathcal{L}$ is a mapping τ from $\mathcal{L}$ to $2^{\mathcal{P}(\mathcal{Q})^+}$, where $\mathcal{P}(\mathcal{Q})^+ = \mathcal{P}(\mathcal{Q}) \setminus \emptyset$.

Definition 5.4.12 Let $G_1, \ldots, G_n(n \in N)$ be soliton graphs and for each $i \in [n]$ let $\mathcal{A}_i$ denote the soliton automaton associated with G_i, i.e. $\mathcal{A}_i = \mathcal{A}(G_i)$ with transition function δ_i. Furthermore, let $\mathcal{L} = \{\mathcal{A}_{n+1}, \ldots, \mathcal{A}_m\}(n \leq m)$ be a system of (not necessarily soliton) automata and τ be a canonical dependency from $\mathcal{Q} = \{\mathcal{A}_1, \ldots, \mathcal{A}_n\}$ to $\mathcal{L}$. A *canonical product* from $\mathcal{Q}$ to $\mathcal{L}$ with respect to τ is a two-level ε-product from $\mathcal{Q}_e = \{\mathcal{A}_e(G_1), \ldots, \mathcal{A}_e(G_n)\}$ to $\mathcal{L}$ with alphabet $X \times X$ and feedback function $\phi = (\phi_1, \ldots \phi_m)$ such that the following conditions hold.

(a) $X = Ext(G_1) \sqcup \ldots \sqcup Ext(G_n)$.

(b) For each $i \in [n]$ and $v, w \in X$

$$\phi_i((v,w)) = \begin{cases} (v,w), & \text{if } v, w \in Ext(G_i) \\ \varepsilon, & \text{otherwise} \end{cases}$$

(c) For any $n+1 \leq i \leq m$, $M_1 \in S(G_1), \ldots, M_n \in S(G_n)$, and $v, w \in X$ such that $v \in Ext(G_k)$ and $w \in Ext(G_l)$ for some $k, l \in [n]$
$\phi_i(M_1, \ldots, M_n, (v,w)) = \varepsilon$ if and only if one of the following conditions holds:
(c/1) $\tau(\mathcal{A}_i) \cap \mathcal{P}_{G_k}(M_k, v) = \emptyset$.
(c/2) $k \neq l$
(c/3) $k = l$, $v \neq w$ and $\delta_k(M_k, (v,w)) = \{M_k\}$.

Observe that if in Definition 5.4.12 the alphabet of each automaton in $\mathcal{L}$ is singleton, then the canonical product is uniquely determined. Indeed, canonical products with the same parameters can differ only in c/3, which is unambiguous in the above case.

Roughly speaking, in a canonical product the automata of $\mathcal{L}$ on the second level are connected to the soliton automata of $\mathcal{Q}$ on the first level, through their canonical classes, according to the canonical dependency τ. State transition is induced in an automaton $\mathcal{A}_j$ in $\mathcal{L}$, only if $\mathcal{A}_j$ is "reachable" in the current state from the first component of the input pair through an appropriate canonical class P_i determined by τ, and if the current input pair also induces a state transition in the automaton containing P_i.

Example 5.4.13 Consider the soliton automata $\mathcal{A}(G_1)$ and $\mathcal{A}(G_2)$ with their states shown in Figure 5.9. It is easy to see that the canonical classes are $P_1 = \{x_1, z_1\}$, $P_2 = \{y_1\}$, $P_3 = \{y_2\}$, $P_4 = \{x_2\}$, and $P_5 = \{z_2\}$. Furthermore, let $\mathcal{A}_3$ denote the full automaton with two states. Define the canonical dependency τ from $\mathcal{Q} = \{\mathcal{A}(G_1), \mathcal{A}(G_2)\}$ to $\mathcal{L} = \{\mathcal{A}_3)$ by $\tau(\mathcal{A}_3) = \{P_1, P_4\}$. Then it is easy to check that there exists a soliton isomorphism between the canonical product from $\mathcal{Q}$ to $\mathcal{L}$ with respect to τ and the soliton automaton in Figure 5.10.

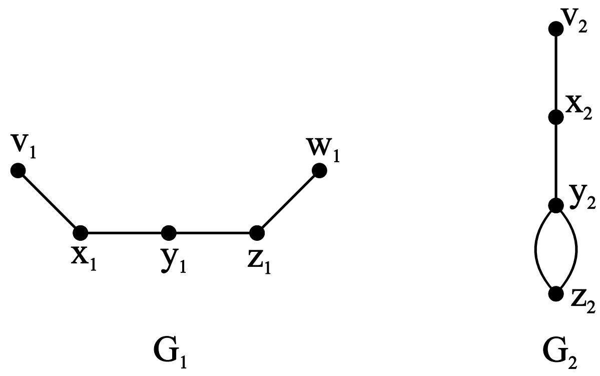

Figure 5.9: Graphs G_1 and G_2 for Example 5.4.13.

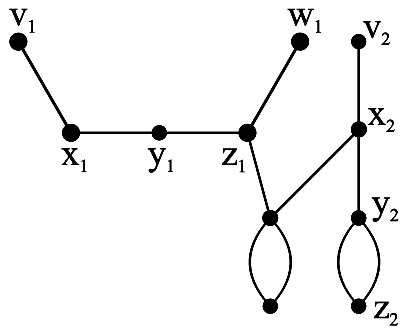

Figure 5.10: The underlying graph of a soliton automaton equivalent by soliton isomorphism to the canonical product of Example 5.4.13.

We will see that the situation captured in Example 5.4.13 generally holds. More precisely: The class of soliton automata is characterized by the canonical products from elementary soliton automata to full automata up to soliton isomorphism. For this goal, first we give the decomposition of soliton automata into elementary ones by canonical product.

Theorem 5.4.14 *Let G be a soliton graph, and let $C_1, \ldots, C_n$ denote the external elementary components of G_h, while $C_{n+1}, \ldots, C_m$ denote the viable internal elementary components of G. Furthermore, for each $n+1 \leq j \leq m$, let $\mathcal{A}_j$ denote the full automaton with $|S(C_j)|$ number of states and define the canonical dependency τ from $\mathcal{Q} = \{\mathcal{A}(C_1), \ldots, \mathcal{A}(C_n)\}$ to $\mathcal{L} = \{\mathcal{A}_{n+1}, \ldots, \mathcal{A}_m\}$ by $\tau(\mathcal{A}_j) = \mathcal{R}_G(C_j)$ with $n+1 \leq j \leq m$. Then $\mathcal{A}(G) \cong_s \mathcal{A}$, where $\mathcal{A}$ is the canonical product from $\mathcal{Q}$ to $\mathcal{L}$ with respect to τ.*

Proof. For each $n+1 \leq j \leq m$, let C'_j be a soliton graph obtained from C_j by attaching a new external edge to some vertex of C_j, and let $\mathcal{A}(C'_j) = (S(C_j), \{x_j\} \times \{x_j\}, \delta^j)$ be the essentially elementary soliton automaton associated with C'_j. Then define the canonical dependency τ' from $\mathcal{Q}$ to $\mathcal{L}' = \{\mathcal{A}_e(C'_{n+1}), \ldots, \mathcal{A}_e(C'_m)\}$ by $\tau'(\mathcal{A}_e(C'_j)) = \mathcal{R}_G(C_j)$ with $n+1 \leq j \leq m$. By part (iv) of Theorem 5.4.4, the canonical product $\mathcal{A}'$ from $\mathcal{Q}$ to $\mathcal{L}'$ with respect to τ' is strongly isomorphic with $\mathcal{A}$.

Furthermore, construct the graph G' from G by adding a loop around each base vertex of G. Now it is easy to see by Theorem 2.3.2 and by the assumption that each constant external edge is constant, that for any external vertex $v \in Ext(G)$ there is a base vertex accessible from

v in any state M of G, by which an M-alternating v-loop exists in G'. Therefore, by Theorems 5.2.7 and 5.2.3, it is clear that $\mathcal{A}_e(G)$ is strongly isomorphic with $\mathcal{A}(G')$.

Based on the observations of the preceding paragraphs, it is enough to prove that $\mathcal{A}_e(G)$ is strongly isomorphic with $\mathcal{A}'$. For the above goal, let δ_e and δ' denote the transition function of $\mathcal{A}_e(G)$ and that of $\mathcal{A}'$, respectively. Moreover, let $y_1, y_2 \in Ext(G)$ and $M \in S(G^+)$ be arbitrary, such that $y_1 \in V(C_r)$, $y_2 \in V(C_s)$ for some $r, s \in [n]$. Since the mapping

$$\psi(M) = (M_{(C_1)}, \ldots, M_{(C_m)}) \tag{1}$$

is clearly a bijection between $S(G^+)$ and $S(C_1) \times \ldots S(C_m)$, we only have to prove that

$$\{\psi(M^*) \mid M^* \in \delta_e(M, (y_1, y_2))\} = \delta'(\psi(M), (y_1, y_2)) \tag{2}$$

To this end, let $\phi = (\phi_1, \ldots, \phi_m)$ be the feedback function of the canonical product resulting in $\mathcal{A}'$, and for each $i \in [m]$, let z_i denote the value of ϕ_i with respect to M and (y_1, y_2), i.e.

$z_i = \phi_i((y_1, y_2))$, if $i \leq n$

and

$z_i = \phi_i(M_{(C_1)}, \ldots, M_{(C_n)}, (y_1, y_2))$, if $n+1 \leq i \leq m$.

Furthermore, for any $j \in [n]$, let δ^j denote the transition function of $\mathcal{A}(C_j)$.

Then,based on (1) and the definition of α_i^{ε}-products, the followings hold for the right side of (2).

$\delta'(\psi(M), (y_1, y_2)) = \delta'((M_{(C_1)}, \ldots, M_{(C_m)}), (y_1, y_2)) =$
$= \delta_e^1(M_{(C_1)}, z_1) \times \ldots \times \delta_e^m(M_{(C_m)}, z_m)$ (3)

Now, in order to study the transition functions and the left side of (2), we distinguish two cases.

Case 1: No soliton walk exists from y_1 to y_2 with respect to M.

In this case, by Theorem 5.2.3 and by Lemma 5.4.8, it is clear that one of the following conditions holds:

(a) $s \neq r$,

(b) $s = r$, $y_1 \neq y_2$ and $\delta^s(M_{(C_s)}, (y_1, y_2)) = \{M_{(C_s)}\}$,

(c) $s = r$, $y_1 = y_2$, $\delta^s(M_{(C_s)}, (y_1, y_2)) = \emptyset$ and for each $P \in \mathcal{P}(C_s)$, $\mathcal{R}_G(C_s) \cap \mathcal{P}_{C_s}(M_{(C_s)}, y_1) = \emptyset$.

Then, according to the definition of τ' and the properties of canonical products, we obtain that for each $i \in [m]$, either $z_i = (y_1, y_2)$ with $i = s = r$, or $z_i = \varepsilon$ otherwise. Nevertheless, in either case $\delta_e^i(M_{(C_i)}, z_i) = \{M_{(C_i)}\}$ $(i \in [m])$ holds, which results in the followings.

$\{\psi(M^*) \mid M^* \in \delta_e(M, (y_1, y_2))\} = \{\psi(M)\} = \{(M_{(C_1)}, \ldots, M_{(C_m)})\} =$
$= \delta_e^1(M_{(C_1)}, z_1) \times \ldots \times \delta_e^m(M_{(C_m)}, z_m)$ (4)

Now equation (2) is obtained by combining (3) and (4).

Case 2: There is a soliton walk from y_1 to y_2 with respect to M.

In this case, clearly $s = r$ holds. Now let $\mathcal{T}(M, y_1, y_2)$ denote the set of M-transition networks from y_1 to y_2, and let

$\mathcal{T}^*(M, y_1, y_2) = \mathcal{T}(M, y_1, y_2)$, if $y_1 \neq y_2$

$\mathcal{T}^*(M, y_1, y_2) = \mathcal{T}(M, y_1, y_2) \cup \emptyset$, otherwise

Furthermore, for any M-alternating network Γ let $E(\Gamma)$ denote the set of edges contained in some alternating unit of Γ, and for any $i \in [m]$, let Γ_{C_i} denote the restriction of Γ to C_i.

Then, making use of Theorem 5.2.3, we obtain the followings for the left side of (2).

$$\{\psi(M^*) \mid M^* = S(M,\Gamma),\ \Gamma \in \mathcal{T}^*(M,y_1,y_2)\} =$$
$$= \{(S_{C_1}(M_{(C_1)},\Gamma_{C_1}),\dots,S_{C_m}(M_{(C_m)},\Gamma_{C_m})) \mid \Gamma \in \mathcal{T}^*(M,y_1,y_2)\} =$$
$$= \{S_{C_1}(M_{(C_1)},\Gamma^1) \mid E(\Gamma^1) \subseteq E(C_1), \Gamma^1 \in \mathcal{T}^*(M,y_1,y_2)\}\} \times \dots$$
$$\dots \times \{S_{C_m}(M_{(C_m)},\Gamma^m) \mid E(\Gamma^m) \subseteq E(C_m), \Gamma^m \in \mathcal{T}^*(M,y_1,y_2)\} \qquad (5)$$

Now comparing (3) and (5), we conclude that the proof becomes complete, if we show that for any $i \in [m]$,

$$\{\ S_{C_i}(M_{(C_i)},\Gamma^i) \mid E(\Gamma^i) \subseteq E(C_i), \Gamma^i \in \mathcal{T}^*(M,y_1,y_2)\} = \delta_e^i(M_{(C_i)},z_i). \qquad (6)$$

For the above goal let $i \in [m]$ be arbitrary, and consider the following two subcases.

Subcase 2a: C_i is internal, i.e. $n+1 \le i \le m$.

Then applying the definition of canonical products and τ', we have:

$z_i = (x_i, x_i)$, if $C_i \in \mathcal{R}_G(C_s) \cap \mathcal{P}_{C_s}(M_{(C_s)}, y_1)$

and

$z_i = \varepsilon$, otherwise.

In the first case, by part (*iv*) of Theorem 5.4.4, $\delta_e^i(M_{(C_i)},z_i) = S(C_i)$ holds, while in the second case $\delta_e^i(M_{(C_i)},z_i) = \{M_{(C_i)}\}$. Now (6) directly follows from Theorem 2.3.1 and Lemma 5.4.8.

Subcase 2b: C_i is external, i.e. $i \le n$.

Then, applying Definition 5.4.12 again, we obtain that:

$z_i = (y_1, y_2)$, if $i = s$ $(i = r)$

$z_i = \varepsilon$, otherwise

If $i \neq s$, then making use of Claim 4.2.2, it is clear that the left side of (6) is equal to $\{M_{(C_i)}\}$, by which we are ready. In other case – $z_i = (y_1, y_2)$ – by Theorem 5.2.3, $\delta^i(M_{(C_i)}, z_i)$ is equal to the set of states M_i^* for which $M_i^* = S(M_{(C_i)}, \Gamma_i)$ with Γ_i being an $M_{(C_i)}$-transition network from y_1 to y_2 in C_i. Now applying Corollary 4.4.15 we obtain (6), by which the proof is complete. $\diamondsuit$

One can ask, why soliton isomorphism is applied in the above theorem instead of strong isomorphism. The following example in Fig 5.11 shows that Theorem 5.4.14 generally does not hold, if we use strong isomorpism.

Let C_1 denote the external and C_2 denote the internal elementary component of the graph G in Fig 5.11. Considering the state M of G represented by double edges, it is clear that $\delta(M,(v_1,v_1))$ contains a state M' such that $M'_{(C_1)} = M_{(C_1)}$ and $M'_{(C_2)} \neq M_{(C_2)}$. Therefore, in order for $\mathcal{A}(G)$ to be strongly isomorphic with the appropriate canonical product of $\mathcal{A}(C_1) = (S(C_1), X_1 \times X_1, \delta_1)$ and that of the full automaton having two states (same number of states than that of C_2), $M_{(C_1)} \in \delta_1(M_{(C_1)},(v_1,v_1))$ should hold, which is impossible. Consequently, we need to consider $\mathcal{A}_e(C_1)$ for the product, which will imply that in the product automaton the input (v_1,v_1) will induce a transition from M to itself. However, only $\mathcal{A}_e(G)$ has such a transition, which explains the use of soliton isomorphism.

Theorem 5.4.14 reduces the analysis of soliton automata to elementary ones, but it is also a question if an arbitrary canonical product from a system of elementary soliton automata to a system of full automata results in a soliton automaton up to soliton isomorphism. The following proposition gives the answer.

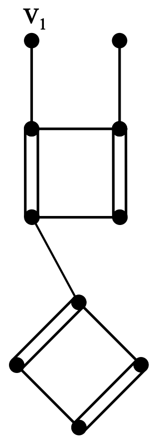

Figure 5.11: Example for soliton automaton demonstrating the use of soliton isomorphism in Theorem 5.4.14.

Proposition 5.4.15 *For any automaton $\mathcal{A}$ obtained by a canonical product from a system of elementary soliton automata to a system of full automata, there exists a soliton graph G such that $\mathcal{A} \cong_s \mathcal{A}(G)$.*

Proof. Let $\mathcal{Q} = \{\mathcal{A}(G_1), \ldots, \mathcal{A}(G_n)\}$ $(n \in N)$ be a system of elementary soliton automata, $\mathcal{L} = \{\mathcal{A}_{n+1}, \ldots, \mathcal{A}_m\}$ $(m \geq n)$ be a system of full automata, and $\mathcal{A}$ be the canonical product from $\mathcal{Q}$ to $\mathcal{L}$ with respect to canonical dependency τ. Furthermore, for each $n+1 \leq i \leq m$, let G_i be a closed elementary graph such that $|S(G_i)|$ is equal to the number of states of $\mathcal{A}_i$. Note that, according to Theorem 5.4.4, G_i exists. Now construct a soliton graph G in such a way that the external elementary components of G are $G_1, \ldots, G_n$, the internal elementary components of G are $G_{n+1}, \ldots, G_m$, and (v, w) is an edge connecting distinct elementary components iff $v \in V(G_i)$ for some $n+1 \leq i \leq m$ and w belongs to a canonical class $P \in \tau(\mathcal{A}_i)$. Consider the canonical product $\mathcal{A}'$ corresponding to the elementary decomposition of G described in Theorem 5.4.14. Then it is easy to see that $G = G_h$, and for each $n+1 \leq j \leq m$, $\mathcal{R}_G(G_j) = \tau(G_j)$ holds. By the above facts it is clear that $\mathcal{A}$ and $\mathcal{A}'$ are strongly isomorphic. Now taking into consideration that $\mathcal{A}(G) \cong_s \mathcal{A}'$, we conclude that $\mathcal{A}(G) \cong_s \mathcal{A}$, as required. $\diamondsuit$

The results of this section are summarized in Theorem 5.4.16. In order to state this theorem, let $\mathcal{S}$ denote the class of automata obtained by a canonical product from a system of elementary soliton automata to a system of full automata.

Theorem 5.4.16 *The class of soliton automata coincides with the class $\mathcal{S}$ up to soliton isomorphism.*

Proof. Immediate by Theorem 5.4.14 and Proposition 5.4.15. $\diamondsuit$

We close this section by a special case of the above theorem, by which a full characterization is obtained for *constant soliton automata* , i.e. soliton automata $\mathcal{A}(G)$ such that each external edge of G is constant. For this, some preparations are needed.

Definition 5.4.17 Let X be an alphabet, and $\mathcal{A}_1, \ldots, \mathcal{A}_m$ $(m > 0)$ be automata. Then for any $k \in N$, an X^k*-product* of $\mathcal{A}_1, \ldots, \mathcal{A}_m$ is a quasi-direct ε-product $\prod_{j=1}^{m} \mathcal{A}_j[X^k, \phi]$ such that for any $x_1, \ldots, x_m \in X$ and $j \in [m]$, $\phi_j((x_1, \ldots, x_m)) \neq \varepsilon$ implies that $x_1 = x_2 = \ldots = x_m$.

Theorem 5.4.18 *Let $\mathcal{C}$ denote the class of automata obtained by an X^2-product of full automata for some alphabet X. Then the class of constant soliton automata coincides with $\mathcal{C}$ up to soliton isomorphism.*

Proof. Let $\mathcal{A}(G)$ be an arbitrary constant soliton automaton and apply Theorem 5.4.14 for $\mathcal{A}(G)$ in order to obtain a canonical product $\mathcal{A} = (S, X \times X, \delta)$ from a system $\mathcal{Q}$ of soliton automata $\mathcal{A}(G_i)$ $(i \in [n], n \in N)$ associated with a mandatory soliton graph to a system $\mathcal{L}$ of full automata $\mathcal{A}_j = (S_j, \{y_j\}, \delta_j)$ $(j = n+1, \ldots, m,\ m \geq n)$ such that $\mathcal{A}(G) \cong_s \mathcal{A}$. Since each automaton of $\mathcal{Q}$ is full with a unique state, either $m = n$ holds, implying that $\mathcal{A}'$ is a full automaton itself, or the projection ϕ from S into $S_{n+1} \times \ldots \times S_m$ is a bijection. In the first case we are ready, whereas in the second case we can construct an X^2-product $\mathcal{A}' = (S', X \times X, \delta')$ of $\mathcal{A}_{n+1}, \ldots, \mathcal{A}_m$ by $S' = S_{n+1} \times \ldots \times S_m$ and by $\delta'(s', (x, y)) = \phi(\delta(\phi^{-1}(s'), (x, y)))$ with $s' \in S'$, $x, y \in X$ such that $\mathcal{A}'$ is strongly isomorphic with $\mathcal{A}$. Therefore we obtain that $\mathcal{A}(G) \cong_s \mathcal{A}'$, by which the theorem is proved in one direction.

Conversely, let $X = \{x_1, \ldots, x_n\}$ $(n \in N)$ be an alphabet, and $\mathcal{A} = (S, X \times X, \delta)$ be an X^2-product of full automata $\mathcal{A}_j = (S_j, \{y_j\}, \delta_j)$ $(j = n+1, \ldots, m,\ m > n)$. Furthermore, for each $i \in [n]$, let G_i be a mandatory elementary soliton graph with $Ext(G_i) = \{x_i\}$ and consider the soliton automaton $\mathcal{A}(G_i) = (\{s_i\}, \{x_i\}, \delta_i)$. Now define the bijection $\phi : \{s_1\} \times \ldots \times \{s_n\} \times S_{n+1} \times \ldots \times S_m \rightarrow S$ by $\phi((s_1, \ldots, s_m)) = (s_{n+1}, \ldots, s_m)$ with $s_j \in S_j$ $(n + 1 \leq j \leq m)$ and apply the construction of the preceding paragraph in reverse in order to obtain an automaton $\mathcal{A}'$ strongly isomorphic with $\mathcal{A}$. It is easy to check that $\mathcal{A}'$ is a canonical product from $\mathcal{Q} = \{\mathcal{A}(G_1), \ldots, \mathcal{A}(G_n)\}$ to $\mathcal{L} = \{\mathcal{A}_{n+1}, \ldots, \mathcal{A}_m\}$, consequently, according to Proposition 5.4.15, there exists a soliton automaton $\mathcal{A}(G)$ such that $\mathcal{A}(G) \cong_s \mathcal{A}'$. By the above facts $\mathcal{A}(G) \cong_s \mathcal{A}$, which implies that $\mathcal{A}(G)$ is a constant soliton automaton. The proof is now complete. $\diamondsuit$

5.5 Elementary Structure Encoding and the Automaton Description Problem for soliton graphs

In this section we return to the problem posed in the Introduction as Question (a).

Automaton Description Problem (ADP): *Given an arbitrary soliton graph G. Give a formal description of the automaton $\mathcal{A}(G)$ associated with G.*

It is clear that ACP is a solution for the above problem, thus we know from Section 5.2 that ADP is effectively computable, and Theorem 5.2.16 provides a bound concerning the algorithmic complexity issues. Nevertheless, the problem has a descriptional complexity aspect too, namely the measure of the information encoding. For the latter question, clearly the transition function gives the upper bound. However, we can learn from Theorem 5.3.9 that both the computational and the descriptional complexity can be significantly reduced by the knowledge of the underlying graph structure. Indeed, considering a soliton graph with one external vertex v, the solution of ADP is simply a natural number representing the number of states and an additional information if the graph is bipartite containing a double v-racket. This observation shows that the complexity gap between the description by transition function and an appropriate encoding can be arbitrarily large. These graphs are also extreme concerning the computational complexity aspects, as the calculation of the state complexity is clearly the minimal task to be carried out for a solution of ADP. In this case the state calculation needs to be complemented only by a method which tests if the given graph is bipartite with a double v-racket. We have seen in Section 5.2 that it can be solved in $\mathcal{O}(n \cdot m)$ time. Unfortunately, counting the number of states of such a graph is NP-hard ([99]), as it is equivalent to determining the number of perfect matchings. Nevertheless, there are important classes of graphs

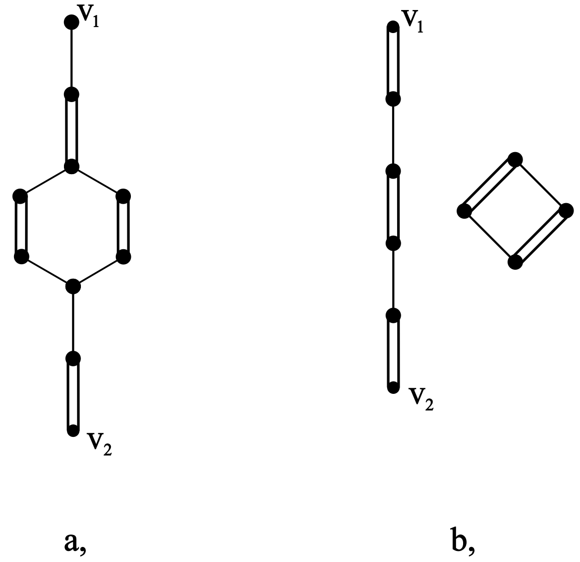

Figure 5.12: Examples for non-deterministically connected vertices.

for which this problem is polynomially computable; probably the most famous one is the class of planar graphs (cf. [70], [71]).

In this section we give a reduction method by which the structure of the graphs can be simplified for ADP. It is a reduction from both computational and descriptional complexity point of view. We will call it Elementary Structure Encoding, as it is based upon the elementary decomposition developed in the previous section. Actually the canonical product described in Theorem 5.4.14 provides almost all the necessary background, the only deficiency that it is expressed by soliton isomorphism in which self-transitions are not captured. Therefore first we give the analysis of self-transitions on the level of elementary soliton automata.

In order to achieve the above goal, for any soliton graph G we will define its self-transition graph G_{self} which is equivalent to G with respect to self-transitions. For this, we need some preparations.

Definition 5.5.1 Let v_1 and v_2 be distinct vertices and M be a state of graph G such that at least one of v_1 and v_2 is internal. We say that v_1 and v_2 are *non-deterministically connected* in state M, if either v_1 and v_2 are connected by two distinct positive M-alternating paths, or there exists an M-alternating cycle which is vertex-disjoint from the unique positive M-alternating path connecting v_1 and v_2.

Figure 5.12 shows a few simple examples for non-deterministically connected vertices.

Proposition 5.5.2 *Let v and w be distinct vertices of a closed graph G having a perfect internal matching. Then v and w are non-deterministically connected in some state of G iff they are such in any state of G.*

Proof. Assume that v and w are non-deterministically connected in state $M \in S(G)$ and M' is a state of G different from M. We need to prove that v and w are non-deterministically connected in M' too. For this goal, construct the graph G_1 by attaching external edges (v, v') and (w, w') to G. Notice that the alternating trails making v and w non-deterministically connected in M become distinct M-alternating units α_1 and α_2 in G_1 such that α_1 is a crossing connecting v' and w', and α_2 is either an alternating cycle disjoint from α_1, or a crossing between v' and w'. Nevertheless, in both cases there exist distinct M-alternating networks Γ_1 and Γ_2 such that $\Gamma_1 = \{\alpha_1\}$, and $\Gamma_2 = \{\alpha_1, \alpha_2\}$ or $\Gamma_2 = \{\alpha_2\}$ depending on whether α_2 is a cycle or a crossing. Since M' is compatible with M, for $i = 1, 2$ the mediator alternating network Γ'_i between M' and $S(M, \Gamma_i)$ also contains an alternating crossing α'_i connecting v' and w'. We know that $\Gamma'_1 \neq \Gamma'_2$, consequently either $\alpha'_1 \neq \alpha'_2$ or there exists an M'-alternating cycle disjoint from α'_1. Now dropping the edges (v, v') and (w, w') from the alternating units above, we obtain the required M'-alternating trails. $\diamondsuit$

By the above result, we can say for vertices v and w of a closed graph that they are *non-deterministically connected* without specifying the state M relative to which this concept was originally defined.

Definition 5.5.3 Let $\{C_1, \ldots, C_n\}$ $(n > 0)$ be the set of the elementary components of soliton graph G with C_1 being its unique external elementary component. Graph G is a *component-chain graph* if it can be decomposed in the *chain-form* $G = C_1 + (w_1, v_2) + C_2 + (w_2, v_3) + \ldots + (w_{n-1}, v_n) + C_n$ such that for each $i \in [n-1]$, $(w_i, v_{i+1}) \in E(G)$ with $w_i \in V(C_i)$ and $v_{i+1} \in V(C_{i+1})$.

With the help of the above concepts now we can introduce G_{self} for any soliton graph G. For this end we need an additional definition.

Definition 5.5.4 Let G be a soliton graph and $v \in Int(G)$. We say that v is an *interlinking vertex*, if v is contained in an external elementary component of G such that v is adjacent to a vertex of an internal elementary component.

For the rest of this section, we drop the assumption held throughout the thesis (cf. Section 2.2) that any internal vertex v of G is augmented automatically by a loop in any subgraph in which v would become external.

Definition 5.5.5 The *self-transition graph* G_{self} of a soliton graph G is constructed in the following way.

For any canonical class P belonging to an external elementary component C of G_h, let $V_P = \bigcup(V(C') \mid C' \in \rho_P((G_h)^+))$, and let G_P denote the graph determined by C, the induced subgraph $V([V_P])$ and the edges connecting a vertex of V_P with a vertex in P. Then $\mathcal{P}_\rho(G)$ denotes the set of canonical classes P contained in an external elementary component of G_h for which G_P satisfies one of the following three conditions:

(a) G_P is not a component-chain graph.

(b) G_P contains a non-bipartite internal elementary component.

(c) G_P is a component-chain graph with chain-form $G_P = C_1 + (w_1, v_2) + C_2 + (w_2, v_3) + \ldots + (w_{n-1}, v_n) + C_n$ $(n > 1)$ such that either v_i and w_i are non-deterministically connected in C_i for some $2 \leq i \leq n-1$ or $C_n + (v_n, w_{n-1})$ is a bipartite graph containing an alternating double w_{n-1}-racket.

Then G_{self} consists of the external elementary components of the graph obtained from G_h by adding a loop around each interlinking vertex belonging to a canonical class contained in $\mathcal{P}_\rho(G)$.

Example 5.5.6 Consider the graph G in Figure 5.13a. Its interlinking vertices are x_1, x_2, y_2, z_2. Applying Definition 5.5.4, it is clear that the unique canonical class P_2 in $\mathcal{P}_\rho(G)$ is the one containing x_2 and z_2. Therefore we obtain the graph represented in Figure 5.13b as G_{self}.

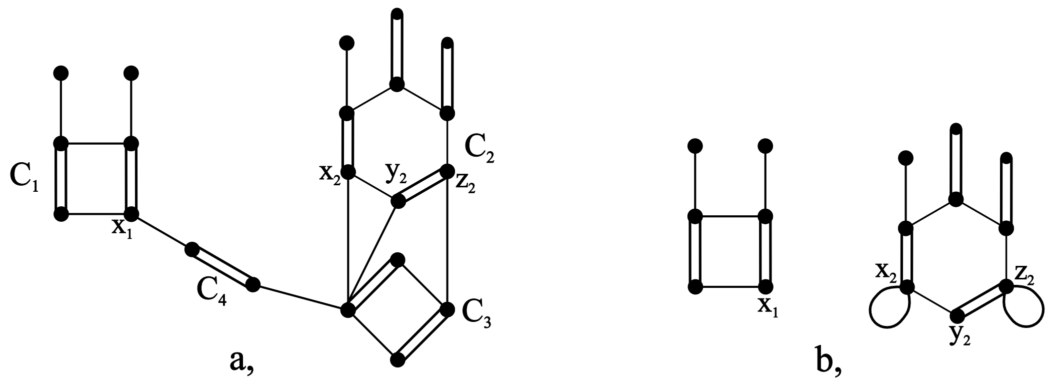

Figure 5.13: Example for defininig the self-transition graph of G.

Now we are ready to prove the result which describes self-transitions on the level of elementary automata.

Theorem 5.5.7 *Let $\mathcal{A}(G) = (S(G^+), X \times X, \delta)$ be a soliton automaton, $v \in X$ be an external vertex of G and $M \in S(G^+)$. Then $M \in \delta(M, (v, v))$ iff one of the following conditions holds for the graph $C[M, v] = C_v^{M(C)}$ constructed from the elementary component C of G_{self} containing v.*

(i) $C[M, v]$ is non-bipartite.

(ii) $C[M, v]$ contains an $M_{C[M,v]}$-alternating double v-racket.

(iii) $C[M, v]$ contains two distinct interlinking vertices being $M_{C[M,v]}$-accessible from v.

(iv) $C[M, v]$ contains an interlinking vertex being non-deterministically connected with v in $M_{(C[M,v])}$.

(v) $C[M, v]$ does not contain an $M_{C[M,v]}$-alternating cycle and there does not exist any interlinking vertex being $M_{C[M,v]}$-accessible from v.

Proof. Since the self-transition graph of G and that of G_h is the same, throughout the proof we can assume by Corollary 5.2.8 that G admits all shortcuts, i.e. $G = G_h$.

Now let C' denote the external elementary component of G corresponding to C, i.e. C' is obtained from C by removing the loops around the interlinking vertices added to C' in the construction of G_{self}. Moreover, let us introduce the shorter notation $C'[M, v]$ for $(C')_v^{M(C')}$, let $\mathcal{C}_v^M = \{C'' \mid C''$ is an internal elementary component of G^+ with $\mathcal{R}_{G^+}(C'') \cap \mathcal{P}_{C'}(M_{(C')}, v) \neq \emptyset\}$, and let $V_{int}(M, v) = \{v \in C'' \mid C'' \in \mathcal{C}_v^M\}$. Then the following holds.

Claim A *G_v^M is equal to the graph determined by $C'[M, v]$ and $G[V_{int}(M, v)]$ together with the edges of G connecting these two subgraphs.*

In order to prove the above statement, first note that, any edge of C' traversed by an M-alternating trail α starting from v will be also contained in the $M_{(C'_h)}$-alternating trail α' obtained from α by making the shortcuts on the C'-loops. Therefore –taking into consideration that $C'_h = C'$–, $E(G_v^M) \cap E(C') = E(C'[M, v])$ holds, consequently we are left to show that $E(G_v^M) \setminus E(C'[M, v])$ consists of the edges having an endpoint in an elementary component belonging to $\mathcal{C}_v^M$.

Now applying Claim 4.2.2, we obtain that any edge of $E(G_v^M) \setminus E(C'[M, v])$ is incident with a vertex contained in an internal elementary component. However, by Lemma 5.4.8, an edge having an endpoint in an internal elementary component is contained in G_v^M iff at least one of its endpoint belongs to a non-principal canonical class of some elementary component in $\mathcal{C}_v^M$. Then the proof of *Claim A* can be finished by observing that, according to Proposition 4.5.24, no edge exists in G^+ connecting two principal vertices.

Also note that $\mathcal{P}_{C'[M,v]}(M_{(C'[M,v])}, v) = \mathcal{P}_{C'}(M_{(C')}, v)$, and $\mathcal{C}_v^M$ consists of the internal elementary components of the graphs G_P (see Definition 5.5.5) with $P \in \mathcal{P}_{C'}(M_{(C')}, v)$.

Suppose first that G_v^M is non-bipartite. Then $M \in \delta(M, (v, v))$ holds by Theorem 5.2.11. We will show that in this case $C[M, v]$ is also non-bipartite. The claim is obvious if α is entirely contained in $C'[M, v]$. Otherwise, let w denote the internal endpoint of α_h, let C_w denote the elementary component containing w, and consider the v-loop α' in G_h constructed from α by making the shortcuts on the C_w-loops being subpaths of α. Since $G = G_h$ holds, α'_c will be an alternating loop in C_w such that if $C_w = C'$, then $E(\alpha') \subseteq E(C'[M, v])$. Therefore α'_c certifies that the edges of $E(G_v^M) \cap E(G_w)$ determine a non-bipartite graph in both cases. Now if $C_w = C'$, then the claim is obvious, whereas if C_w is internal, then by condition (b) of Definition 5.5.5, $\mathcal{R}_{G^+}(C_w)$ will contain a canonical class P of $\mathcal{P}_{C'}(M_{(C')}, v) \cap \mathcal{P}_\rho(G)$, resulting in that $C[M, v]$ is non-bipartite by a loop around a vertex of P.

By the observations of the previous paragraph, the statement of the theorem holds if G_v^M is non-bipartite, thus we can assume for the rest of the proof that G_v^M is bipartite. Clearly, in this case the theorem is equivalent to stating that one of conditions $(i) - (v)$ holds iff either there exists an $M_{(G_v^M)}$-alternating double v-racket or there does not exist any $M_{(G_v^M)}$-alternating cycle.

'Only if' Assume first that there does not exist an $M_{(G_v^M)}$-alternating cycle. Since G_v^M is bipartite, the above fact clearly implies by Claim A that $C'[M, v] = G_v^M$. Then it is easy to see that condition (v) holds, by which we are ready.

Considering the other case let $\alpha = (\alpha^1, \alpha^2)$ be an M-alternating double v-racket with branching vertex w, and for $i = 1, 2$, let α'_i denote the prefix of α_h^i from v to its last vertex x_i belonging to C'. Since G admits all shortcuts, we can suppose that α'_i $(i = 1, 2)$ does not contain any C'-loop as subpath. If α is entirely contained in C', then (ii) holds and we are ready. Otherwise, for some $i \in \{1, 2\}$, say $i = 1$, α'_i is different from α_h^i. Then the followings are easy to be observed:

(a) If α_c^k with $k \in \{1, 2\}$ is contained in C', then (iv) holds by α'_1 and α_c^k.

(b) If $w \in V(C')$ with neither of α_c^1 and α_c^2 being contained in C', then by α'_1 and α'_2, either (iii) or (iv) holds depending on whether $u_1 \neq u_2$ or $u_1 = u_2$.

Based on the above facts we may assume that $\alpha'_1 = \alpha'_2$ and α_c^k $(k = 1, 2)$ is not in C'. Now let P denote the canonical class containing $u_1 = u_2$ and consider the graph G_P introduced in Definition 5.5.5. If G_P is not a component-chain graph then there exists a loop around u_1 in G_{self} making $C[M, v]$ non-bipartite. Therefore, assume that G_P is a component-chain graph with the chain-form $C_1 + (w_1, v_2) + \ldots + (w_{n-1}, v_n) + C_n$ such that $C_1 = C'$. Then, in order to apply condition (iii), we will show that, either v_r and w_r are non-deterministically connected in C_r for some $2 \leq r \leq n - 1$, or $C_n + (v_n, w_{n-1})$ contains an alternating double w_{n-1}-racket.

For the above goal, let C_i and C_j $(i, j \in [n])$ denote the internal elementary components containing α_c^1 and α_c^2, respectively. Furthermore, for $m = 1, 2$ and $l = i, j$, let α_l^m denote the subtrail of α^m running in C_l. We may suppose without loss of generality that $i \leq j$. Now consider the elementary component C_k containing w. If $k < i$, then it is easy to see that v_k and w_k are non-deterministically connected by α_k^1 and α_k^2. Similarly, if $i < j$, then α_c^1 with α_i^2 will make v_i and w_i non-deterministically connected. Therefore we may assume in the followings that $k = i = j$.

Now if $i = n$, then attaching (w_{n-1}, v_n) to α_n^1 and α_n^2, we will obtain an $M_{(C_n)}$-alternating double w_{n-1}-racket, as required. In the other case we will show that v_i and w_i are non-deterministically connected in C_i. To this end first observe that there exists a positive $M_{(C_i)}$-alternating path β_i between v_i and w_i. Indeed, it is easy to see by Corollary 5.4.9, that there exists an external M_{G_P}-alternating path β leading from v to some vertex of C_{i+1}. Therefore the subpath of β running in C_i is a suitable choice for β_i. Now let (A_i, B_i) denote the bipartition of C_i with $w_i \in B_i$ (or equivalently, $v_i \in A_i$), and starting from v_i let u_i denote the last vertex of β_i which is also in $V(\alpha_i^1) \cup V(\alpha_i^2)$. We can assume without loss of generality that $u_i = u$ for some $u \in V(\alpha_i^1)$. Then $\alpha' = \alpha_i^1[v_i, u] + \beta_i[u_i, w_i]$ is a positive $M_{(C_i)}$-alternating path, because u_i must belong to B_i. If α' is disjoint from any of α_c^1 and α_c^2, then we are ready. Otherwise, it is easy to see that $\alpha_c^1 = \alpha_c^2$ holds with $E(\alpha') \cap E(\alpha_c^1) \neq \emptyset$. In this case the first vertex of α_i^2 incident with an edge of $E(\alpha_i^2) \setminus E(\alpha')$ is the branching vertex w, which is clearly different from w_i. Thus continuing α_i^2 from w, there will be a next vertex $u_i' \in V(\alpha_i^2)$ with the property that u_i' is also on α'. We know that $\alpha_h^1[v_i, w]$ is positive at the w end, which implies, with the help of $\alpha_h^1[v_i, w] = \alpha'[v_i, w]$, that $w \in B_i$. Therefore $u_i' \in A_i$, because the other case would result in an alternating loop by $\alpha'[w, u_i'] + (\alpha_i^2)^{-1}[u_i', w]$, which is impossible in a bipartite graph. Summarizing the above facts, we can conclude that α' and $\alpha'[v_i, w] + \alpha_i^2[w, u_i'] + \alpha'[u_i', w_i]$ will provide the required distinct $M_{(C_i)}$-alternating paths, by which the proof of the 'Only if part' is complete.

'If' The statement is trivial, if condition (ii) stands for $C[M, v]$. Furthermore, if condition (v) holds, then it is clear by Claim A that $C[M, v] = G_v^M$, which implies that there does not exist an $M_{(G_v^M)}$-alternating cycle. Therefore we assume for the rest of the proof that one of conditions (i), (iii), and (iv) holds such that $C'[M, v]$ does not contain an $M_{C'[M,v]}$-alternating double v-racket. For the analysis of the different cases, we will use the following two observations.

Claim B *For any internal elementary component $C'' \in \mathcal{C}_v^M$, $\mathcal{F}_{C''}$ is an internal family consisting of $C'' = r(\mathcal{F}_{C''})$.*

For the proof of Claim B, suppose on the contrary that $\mathcal{F}_{C''}$ contains a two-way elementary component C_1. Notice that, in this case $C_1 \in \mathcal{C}_v^M$ holds. Indeed, if $\mathcal{F}_{C''}$ was external, then C'' could be chosen as C_1, while in other case Proposition 5.4.7, part (c) would imply the above fact. Therefore, according to Lemma 5.4.8, the endpoints of any edge in C'' are M-accessible from v, which is a contradiction by Proposition 5.2.9.

Claim C *Let $P \in \mathcal{P}_{C'}(M_{(C')}, v)$ be a canonical class, and let $C'' \in \rho_P(G^+)$ such that for any elementary component $C_1 \in \rho_P(G^+)$, $\mathcal{F}_{C''} \not\mapsto \mathcal{F}_{C_1}$ holds. Then C'' is a non-mandatory elementary component.*

The above statement easily follows from the assumption that G_v^M is non-bipartite. Indeed, in that situation C'' does not contain a loop and by Claim B it is one-way. Consequently, if C'' was mandatory, then its non-principal vertex should be connected to another elementary component. Now combining the preceding observations with the obvious fact that C'' is a minimal element by the partial order $\overset{*}{\mapsto}$, we obtain Claim C.

Consider now the set $\mathcal{P}_{int}$ consisting of the canonical classes in $\mathcal{P}_{C'}(M_{(C')}, v)$ which contain an interlinking vertex. Observe that by the assumption made at the beginning of the proof of the 'If' part, $\mathcal{P}_{int}$ is nonempty. Indeed, if $\mathcal{P}_{int}$ was empty, then $G_v^M = C[M, v]$ would hold, which means – since G_v^M is bipartite – that either of conditions of (ii) and (v) would be fulfilled, which is a contradiction.

By the previous paragraph, $|\mathcal{P}_{int}| \geq 1$. Now suppose first that $|\mathcal{P}_{int}| > 1$, i.e. $\mathcal{P}_{int}$ contains two distinct canonical classes P_1 and P_2. In that case, for an elementary component $C_i \in \rho_{P_i}(G^+)$ $(i = 1, 2)$ with $\mathcal{F}_{C_i}$ being minimal by the partial order $\overset{*}{\mapsto}$ (such a component exists by the recursive construction in Definition 5.4.6), according to Claim C, C_i contains an M-alternating cycle β_i. Moreover, by Corollary 5.4.9, for $i = 1, 2$ there exists an M-alternating

v-racket γ^i such that $\gamma^i_c = \beta_i$ and the last vertex of γ^i_h common with C' belongs to P_i. The above facts imply that $\gamma^1_h \neq \gamma^2_h$, by which if $\beta^1 = \beta^2$, then (γ^1, γ^2) is an M-alternating double v-racket. Otherwise, i.e. $C_1 \neq C_2$, notice that $E(\gamma^1_h) \cap E(\gamma^2_c) = \emptyset$ and $E(\gamma^2_h) \cap E(\gamma^1_c) = \emptyset$. Indeed, assuming that e.g. $E(\gamma^1_h) \cap E(\gamma^2_c) \neq \emptyset$, we would obtain by Corollary 4.5.23 that $\mathcal{F}_{C_2} \overset{*}{\mapsto} \mathcal{F}_{C_1}$, which is a contradiction in the choice of $\mathcal{F}_{C_1}$ and $\mathcal{F}_{C_2}$. Therefore in this case (γ^1, γ^2) constitute an M-alternating double v-racket, again.

By the previous paragraph we may assume for the rest of the proof that $\mathcal{P}_{int}$ consists of a unique canonical class P. Furthermore, as a first subcase, suppose that condition (iv) holds. Then, making use of Claim C and Corollary 5.4.9, we obtain that there exists an M-alternating v-racket γ and an $M_{(C'[M,v])}$-alternating trail β being edge-disjoint from γ such that γ_c is contained in an internal elementary component, and β is either an alternating cycle or a negative alternating path connecting two vertices of γ_h. Now a suitable double v-racket $\delta = (\gamma, \gamma')$ can be constructed in such a way that in the first case $\gamma'_c = \beta$ and γ'_h is a suitable $M_{(C'[M,v])}$-alternating path α connecting v and some vertex of β (α exists by Claim A), while in the second case, $\gamma'_c = \gamma_c$ and, utilizing that G^M_v is bipartite, γ_h is determined by β and by the appropriate subpaths of γ_h.

Assume now that $C[M, v]$ is non-bipartite. In the present situation it is clearly equivalent to saying that either condition (a) or (c) of Definition 5.5.5 holds for the graph G_P. Suppose first that G_P is not a component-chain graph. Then G_P has two distinct subgraphs G_1, G_2 having the property that G_i $(i = 1, 2)$ is a component-chain graph with chain-form $G_i = C^i_1 + (w^i_1, v^i_2) + C^i_2 + (w^i_2, v^i_3) + \ldots + (w^i_{n_i-1}, v^i_{n_i}) + C^i_{n_i}$ $(n_i > 1)$ such that $C^i_1 = C'$, $C^i_{n_i}$ is minimal by $\overset{*}{\mapsto}$, and for $j \in [n_i - 1]$, $\mathcal{F}_{C^i_j} \mapsto \mathcal{F}_{C^i_{j+1}}$. Now combining Corollary 5.4.9 with Claim C again, we obtain that for $i = 1, 2$, $C^i_{n_i}$ contains an $M_{(C^i_{n_i})}$-alternating cycle β_i, and there exists an M_{G_i}-alternating v-racket γ^i such that $\gamma^i = \beta_i$. Since G_1 and G_2 are distinct component-chain graph, it is clear that $\gamma^1_h \neq \gamma^2_h$. Furthermore, as we have noticed in a similar situation, for $i = 1, 2$, $E(\gamma^i_h) \cap E(\gamma^{3-i}_c) = \emptyset$ follows from the fact that $C^i_{n_i}$ is minimal by $\overset{*}{\mapsto}$. Summarizing the above observations, we conclude that (γ^1, γ^2) is an M-alternating double v-racket, as required.

Now suppose that G_P is a component-chain graph with chain-form $G_P = C_1 + (w_1, v_2) + C_2 + (w_2, v_3) + \ldots + (w_{n-1}, v_n) + C_n$ $(n > 1)$ such that $C_1 = C'$ and G_P satisfies condition (c) in Definition 5.5.5. It is clear by the definition of $\rho_P(G_P)$, that for $j \in [n-1]$, $\mathcal{F}_{C_{j-1}} \mapsto \mathcal{F}_{C_j}$. Consequently, as we have seen several times, Corollary 5.4.9 and Claim C imply that there exists an M-alternating v-racket β such that β_c is contained in C_n. Below in each subcase we will give the construction of an M-alternating double v-racket $\delta = (\delta^1, \delta^2)$.

If $C_n + (v_n, w_{n-1})$ contains an alternating double w_{n-1}-racket γ, then we are ready by $\delta^1_c = \gamma^1_c$, $\delta^2_c = \gamma^2_c$, $\delta^1_h = \beta_h[v, w_{n-1}] + \gamma^1_h$ and $\delta^2_h = \beta_h[v, w_{n-1}] + \gamma^2_h$. Otherwise, let C_i $(2 \leq i \leq n-1)$ denote the elementary component in which v_i and w_i are non-deterministically connected. In that case there exists an $M_{(C_i)}$-alternating trail α being edge-disjoint from β such that α is either an alternating cycle or a negative alternating path connecting two vertices of β_h. Furthermore, if α is a cycle, then it traverses a vertex w with $w \not\sim v_i$, implying that w is connected with v_i by a positive $M_{(C_i)}$-alternating path α_i. Therefore, in this case $\delta^1 = \beta$, $\delta^2_c = \alpha$, and δ^2_h is the appropriate prefix of $\beta_h[v, v_i] + \alpha_i$. Finally, if α is a negative alternating path connecting two vertices of β_h, then taking advantage of C_i being bipartite, we obtain that joining α with the appropriate subpaths of β_h will result in an external negative M-alternating path β'. Then δ is given by $\delta^1 = \beta$, $\delta^2_c = \beta_c$ and $\delta^2_h = \beta'$.

In order to finish the proof, we are left the case when condition (iii) holds with $\mathcal{P}_{int} = \{P\}$. However, the above fact implies that G_P is not a component-chain graph, which situation has been already analyzed. Therefore the proof is complete. $\diamondsuit$

The above result has a simplified form for constant external edges.

Corollary 5.5.8 *Let $\mathcal{A}(G) = (S(G), X \times X, \delta)$ be a soliton automaton and let G_{self} be its self-transition graph. Furthermore, let $v \in X$ be an external vertex of G such that the external edge incident with v is constant. Then the following three statements are equivalent.*

(i) $M \notin \delta(M, (v, v))$ for some state M of G.

(ii) $M \notin \delta(M, (v, v))$ for all states M of G.

(iii) The elementary component C of G_{self} containing v is loop-free, i.e. consists of a single edge.

Proof. Using the notations of Theorem 5.5.7, it is clear that $C[M, v] = C$ for any state M and only condition (i) can be held for C. Now the proof is straightforward by Theorem 5.5.7 and by the observation that the construction of G_{self} is matching-invariant. ♢

The above result made complete the arsenal to define our structure encoding. For this goal, let G be an arbitrary soliton graph and apply the following method.

1. Isolate the elementary components of G together with its canonical classes and construct G_h. Then determine $(G_h)^+$, construct the set $\rho_P(G_h^+)$ for each canonical class P belonging to an external elementary component of G, and based on this, determine the canonical dependency τ according to Theorem 5.4.14.
2. Construct the self-transition graph G_{self} with its interlinking vertices $V_{link}(G)$.
3. For each internal elementary component C_i determine $|S(C_i)|$ and define a code c_i such that $c_i = |S(C_i)|$. Let $\mathcal{C}_{int}(G)$ denote the set of these codes. Define the mapping τ_c from $\mathcal{C}_{int}(G)$ to the power set of the canonical classes of G_{self} such that $\tau_c(c_i) = \tau(C_i)$.

The above method is called the *Elementary Structure Encoding*, while the resulted code $\mathcal{E}(G) = (G_{self}, \mathcal{P}(G_{self}), V_{link}(G), \mathcal{C}_{int}(G), \tau_c)$ is referred to as the *Elementary Structure Code* of G.

It is clear that $\mathcal{E}(G)$ together with Theorems 5.4.14 and 5.5.7 provide a reduction on ADP. Indeed, by this encoding, the analysis of soliton automata is reduced to elementary automata both from theoretical and practical point of view. We have seen that ADP strongly depends on the structure of the underlying graph. Nevertheless, our encoding contains all the necessary information for the analysis, thus it is a real improvement concerning descriptional complexity. The information can be obtained efficiently from this structure, and concerning the operation of the automaton, it is enough to investigate the automata associated with the external elementary components. Comparing to ACP, this encoding provides a significant reduction from algorithmic aspects, as the internal part is fully described by the state calculation and the dependency relation.

Now we will analyze the computational complexity of the construction of our structure code.

Theorem 5.5.9 *Let G be a soliton graph and $n = |V(G)|$. Then the elementary decomposition with the canonical partition of $(G_h)^+$ can be determined in $\mathcal{O}(n^3)$ time.*

Proof. It is easy to see that we need to solve the following problem in order to determine both the elementary decomposition and the canonical partition.

Problem 1 Given an internal vertex w. Determine the set of internal vertices which attract w.

By answering this question we can decide for each edge if it is allowed, which gives the elementary decomposition. Once the elementary decomposition is obtained, the intersection the above sets with the vertex sets of the elementary components results in the canonicl partition.

For solving *Problem 1*, construct the closure G^* of G. By Corollary 4.3.5, any internal vertex u attracts w in G iff u attracts w in G^*. Now attach an external edge (v, w) to w and apply the algorithm of Section 4.7 in order to determine the vertices accessible from v. By Lemma 3.2.7 these vertices exactly the ones which attract w. Therefore *Problem 1* can be solved in $\mathcal{O}(|E(G^+)|)$ time for each vertex w.

We also need to identify the hidden edges. By Lemma 4.4.6 and Theorem 4.3.7, we can carry out the above task in G^*. Then for each internal vertex w apply the following method.

Consider the elementary component C containing w in G^*, and take the graph G' obtained from G^* by deleting all the edges of C. If w is isolated in this graph, then there is no C-loop starting from w. Otherwise, attach a new external edge (u', u) to each non-isolated vertex u of C such that $u' \notin V(G^*)$. Now extend the given matching to the new external edges to obtain a perfect matching, and add a loop around each new vertex u', except the one adjacent to w. Now apply again the algorithm of Section 4.7 in the resulted graph, in which w is the unique base vertex. Now it is clear that a C-loop connects w with a vertex x of $V(C)$ iff the extra vertex attached to x is accessible in this graph.

Now the viable part $(G_h)^+$ of G_h can be obtained in $\mathcal{O}(m)$ time by the algorithm of Section 4.7, where m denotes the number of edges of G_h.

Finally, we must determine the running time of the algorithm. Observe that each of the above methods applies the algorithm of Section 4.7 at most n times, by which the complexity is $\mathcal{O}(n \cdot |E(G^*|)$. However, because of the marginal edges, $|E(G^*| = \mathcal{O}(n^2)$, which proves the statement concerning complexity. $\diamondsuit$

Theorem 5.5.10 *For any soliton graph G, its self-transition graph G_{self} can be constructed in $\mathcal{O}(n^3)$ time, where n denotes the number of vertices of G.*

Proof. We can suppose that the elementary decomposition and the canonical partition of $(G_h)^+$ is given by Theorem 5.5.9. Now studying Definition 5.5.5, the non-trivial parts of the calculation are: determining $\rho_P((G_h)^+)$ for any canonical class P belonging to an external elementary component, testing the existence of a double w-racket in an essentially internal elementary graph, and deciding for certain pair of vertices that they are non-deterministically connected in an internal elementary component.

The solution of the first problem is straightforward by Phase 2 of the algorithm of Section 4.7 and by Proposition 5.4.7. It runs in $\mathcal{O}(m)$ time, if m denotes the number of edges of G_h. The algorithm for the second problem is given by Algorithm 5.2.13 and its complexity is $\mathcal{O}(n \cdot m)$ by Theorem 5.2.14 considering all the elementary components for which it was applied.

Finally, if C is an internal elementary component, v and w are vertices for which we want to test if they are non-deterministically connected, then a naive method is the following. Determine a positive alternating path α between the given vertices. It can be carried out in $\mathcal{O}(m)$ time by our alternating path method used several times. Now delete the edges of α from C. If the remaining graph contains an alternating cycle (which can be tested by the method of [51]), then we are ready. Otherwise, for each inner vertex $u \in V(\alpha)$, check if v and w are connected by a positive alternating path in $C - u$. It is clear that there exists an alternating path between v an w in C which is different from α iff they are connected by an appropriate alternating path in $G - u$ for some vertex of α. As we use our $\mathcal{O}(m)$ complexity algorithm at most once for each vertex, the running time of this phase is bonded by $\mathcal{O}(n \cdot m)$.

Therefore we obtained that the complexity of the above methods is $\mathcal{O}(n \cdot m)$ which is together with the necessary procedures of Theorem 5.5.9 result in an algorithm with $\mathcal{O}(n^3)$ complexity, as expected. $\diamondsuit$

By the above algorithms the first two steps of the method of Elementary Structure Encoding can be computed efficiently, which results in the following theorem concerning the algorithmic aspects of the construction of our code.

Theorem 5.5.11 *For any soliton graph G, $\mathcal{E}(G)$ can be computed in polynomial time iff the number of states of the internal elementary components of G^+ can be counted in polynomial time.*

As we mentioned earlier, the state complexity calculation is necessary for ADP, thus better reduction is not expected on this level. Nevertheless, we have large classes of graphs for which the above problem is provable polynomial. One of the most classical results in matching theory is from Kasteleyn (cf. [70], [71]), who proved that the number of perfect matchings can be counted in polynomial time for any planar graph. This theorem has been extended for $K_{3,3}$-free graphs by Little ([82]), which is indeed a generalization in the light of Kuratowski's classical result([81]).

Theorem 5.5.12 *Let G be a soliton graph such that each internal elementary component in G^+ is a planar graph. Then $\mathcal{E}(G)$ can be computed in polynomial time.*

Summarizing the results of this chapter, we can conclude that if given a soliton graph G, then the complexity bound for ADP is determined by the efficiency of ACP for the external components and the complexity of counting the states of the internal elementary components. Therefore, combining our structure encoding with the results for ACP, we obtain a sufficient condition for ADP to be polynomially solvable.

Theorem 5.5.13 *Let G be a soliton graph such that each of its external elementary components has a polynomial number of states and the state complexity of each internal elementary component of G^+ can be determined in polynomial time. Then ADP can be solved in polynomial time for G.*

Proof. Immediate by Corollary 5.2.17, Theorem 5.5.11, and by the Elementary Structure Encoding. $\diamondsuit$

Chapter 6

Deterministic soliton automata

6.1 Introduction

In the analysis of complex systems it is a central question to describe the characteristics which make a given system deterministic. Therefore both from theoretical and practical point of view, the characterization of deterministic soliton automata is a fundamental problem. Naturally, the above goal can be achieved only through the description of the structure of deterministic soliton graphs.

The operation of the internal part of any soliton automaton is captured by Theorem 5.4.14, hence it is an obvious generalization of determinism to introduce partially deterministic soliton automata as automata associated with a graph such that each of its external elementary components is deterministic. This extension of determinism is also motivated practically by the structure encoding developed in the previous chapter, because the complexity of ADP for deterministic and partially deterministic automata may differ only in calculating the state number of the internal elementary components.

In this chapter we will describe the class of deterministic and partially deterministic soliton automata. In order to achieve the above goal first we give the matching-theoretic characterization of deterministic soliton graphs: in Section 6.2 it is proved that a viable soliton graph is deterministic if and only if each of its connected components are either a chestnut or a graph without alternating cycle. Nevertheless, this result does not provide a structural description of deterministic soliton graphs. Furthermore, on the basis of the above result, one cannot design an efficient algorithm for checking the determinism of a given graph, as one should test the existence of an alternating cycle in all states, the number of which might be exponential. Therefore, making use of the closing results of Section 6.2 describing the class of deterministic and partially deterministic soliton automata by canonical and disjoint products of deterministic elementary soliton automata, the rest of the chapter concentrates on the analysis of elementary soliton graphs without alternating cycles.

To reach the above goal, in Section 6.4 a structural characterization is given for deterministic elementary soliton graphs, which is based on the reducing procedure introduced in Section 6.3. This reducing technique is actually an extension of the one studied in Section 2.4 which technique eliminates the loops around vertices with degree greater than 3. It is proved that any soliton graph has a minimal representation with respect to the reducing procedure, and the automaton associated with the minimal representation of any deterministic soliton graph is strongly isomorphic with the original automaton. We also prove two observations on reduced generalized trees, i.e. graphs without even-length cycles to which no reducing operation can be applied.

The main result of this chapter is presented in Section 6.4 stating that a non-mandatory elementary soliton graph is deterministic iff it can be reduced to a generalized tree. This result leads then a to more sophisticated description of the class of deterministic and partially deterministic soliton automata by products of reduced generalized trees and that of baby chestnuts (chestnuts consisting of two parallel edges and a number of external edges having their internal endpoints in common):

(i) Let $\mathcal{T}$ denote the class of soliton automata associated with either a reduced generalized tree or mandatory elementary graph. Then the class of partially deterministic soliton automata and the class of automata obtained by a canonical product from a system of soliton automata in $\mathcal{T}$ to a system of full automata coincide up to soliton isomorphism.

(ii) Let $\mathcal{D}$ denote the class of soliton automata $\mathcal{A}(G)$ such that either $\mathcal{A}(G)$ belongs to $\mathcal{T}$ or G is associated with a baby chestnut. Then the class of deterministic soliton automata and the class of automata obtained by a disjoint product of soliton automata in $\mathcal{D}$ coincide up to strong isomorphism.

Finally, the above characterization of the graph structure of reduced deterministic elementary graphs results in a polynomial time algorithm deciding if a graph is deterministic. This algorithm consists of three methods: the construction of the elementary decomposition of the given soliton graph, the reduction procedure for the external elementary components, and a method testing the existence of a cycle of even length in the reduced external elementary components.

Section 6.2 is from [14] and has not been published earlier. Sections 6.3 and 6.4 are based on the papers [18] and [78]. In [78] the reduction operation of Section 2.4 was applied to 1-extendable graphs only, and it was proved that a 1-extendable graph reduced with respect to the above operation is deterministic iff it is a generalized tree. The algorithm of Section 6.4 was also presented in this paper. The discussion of [18] goes beyond the characterization given in [78] by extending the scope of investigations from 1-extendable to all elementary soliton graphs, and by elaborating a linear algorithm to decide if an arbitrary graph G is a deterministic elementary soliton graph.

6.2 Matching-theoretic characterization of deterministic soliton graphs

In this section we will characterize the deterministic property of soliton graphs by graph matchings. To this end first we prove two important properties of such graphs.

Lemma 6.2.1 *Let G be a deterministic soliton graph, β be an alternating cycle with respect to some state M of G, and v be an external vertex from which β is M-accessible. Then the external edge e incident with v is constant.*

Proof. Suppose on the contrary that e is non-constant. Then, by Theorem 2.3.2, there exists an M-alternating crossing α connecting v with some external vertex w. Observe that α intersects β. If α and β are vertex-disjoint, then both $\Gamma_1 = \{\alpha\}$ and $\Gamma_2 = \{\alpha, \beta\}$ are M-transition networks from v to w. Therefore, by Theorem 5.2.3, in this case $S(M, \Gamma_1), S(M, \Gamma_2) \in \delta(M, (v, w))$ holds for the transition function δ of $\mathcal{A}(G)$, which is a contradiction.

By the preceding paragraph, we can assume that α intersects β. Consider now the mediator alternating network Γ between $S(M, \beta)$ and $S(M, \alpha)$. Γ will contain a crossing α' different from α, yet connecting the same two external vertices v, w. Thus, for the state $M' = S(M, \alpha)$, $\mathcal{A}(G)$ has two different transitions on input (v, w) resulting in states $S(M', \alpha')$ and $S(M', \alpha) = M$, respectively. Hence, we obtained a contradiction again, by which the proof is complete. $\diamondsuit$

Lemma 6.2.2 *Let G be a viable deterministic soliton graph, v be an external vertex incident with a constant edge e and C be the external elementary component containing v. Then there exists at most one non-mandatory internal elementary component C' such that $\mathcal{F}_C \overset{*}{\mapsto} \mathcal{F}_{C'}$. Furthermore, if an elementary component C' with the above conditions exists, then the followings hold.*

(i) C' consists of a unique even cycle.

(ii) $\mathcal{F}_{C'} = \{C'\}$ and $\mathcal{F}_{C'}$ is a minimal element by the partial order $\overset{*}{\mapsto}$.

(iii) $\mathcal{F}_C$ is either degenerate or consists of a loop-free mandatory elementary component.

(iv) If $\mathcal{F}_1$ is an internal family such that $\mathcal{F}_C \overset{*}{\mapsto} \mathcal{F}_1$ with $\mathcal{F}_1 \neq \mathcal{F}_{C'}$, then $\mathcal{F}_1$ consists of a unique loop-free mandatory elementary component.

(v) For any family $\mathcal{F}_1$ such that $\mathcal{F}_C \overset{*}{\mapsto} \mathcal{F}_1$ and $\mathcal{F}_1 \neq \mathcal{F}_{C'}$, there exists a unique family $\mathcal{F}_2$ with $\mathcal{F}_1 \mapsto \mathcal{F}_2$, and $\mathcal{F}_1$ is connected with $\mathcal{F}_2$ by a single edge.

Proof. As in Chapter 5, we can assume without loss of generality that e is mandatory. Let M be an arbitrary state of G and consider the automaton $\mathcal{A}(G) = (S(G^+, X \times X, \delta)$ associated with G. Now if α is an M-alternating cycle accessible from v in M, then by Theorem 5.2.3, $S(M, \alpha) \in \delta(M, (v, v))$ holds. Since $\mathcal{A}(G)$ is deterministic, the above facts imply the following claim.

Claim A There exists at most one M-alternating cycle which is accessible from v in M.

Furthermore, combining the remarks preceding Claim A with Proposition 5.2.10 and Theorem 5.2.7, we obtain another important observation:

Claim B If there exists an M-alternating cycle being M-accessible from v, then G_v^M is bipartite.

It is clear by Corollary 5.4.10 that G_v^M is determined by those edges having at least one endpoint in a non-principal canonical class of some elementary component belonging to the set $\mathcal{C}_v = \{C_1 \mid \mathcal{F}_C \overset{*}{\mapsto} \mathcal{F}_{C_1}\}$. However, G is viable, thus Proposition 4.5.24 implies that G does not contain edges connecting two principal vertices. Summarizing the above observations we have:

Claim C G_v^M is equal to the subgraph induced by the vertices of the elementary components in $\mathcal{C}_v$.

Now it is clear by Claims A and C that there is indeed at most one non-mandatory elementary component C' in $\mathcal{C}_v$; as expected. Then let C' denote such a component for the rest of the proof, and let α be an M-alternating cycle of C'. In order to prove (i), suppose on the contrary that there exists an edge $e \in E(C')$ which is not contained in α. Combining Proposition 4.6.2 with Claims B and C, we can conclude that e is allowed. However, in this case by Theorem 2.3.2, an $M_{(C')}$-alternating cycle β would cross e, implying that both α and β are M-accessible from v. Therefore a contradiction with Claim A has been established, by which (i) is proved.

Apply again Proposition 4.6.2 together with Claims B and C. Now we obtain for any elementary component $C_1 \in \mathcal{C}_v$ that $\mathcal{F}_{C_1} = \{C_1\}$, and if in addition, $C_1 \neq C'$, then C_1 is a loop-free mandatory component. The above situation easily implies that among the members of $\mathcal{C}_v$, C' is the unique minimal element by $\overset{*}{\mapsto}$. Then it is clear that $(ii) - (iv)$ are proved by the observations of this paragraph.

Finally, in order to verify statement (v), suppose on the contrary that G_v^M is not a component-chain graph. In this case, by Definition 5.5.5 and Corollary 5.5.8, we obtain that $M \in \delta(M, (v, v))$. However, we have seen for the alternating cycle α of C' that $S(M, \alpha) \in \delta(M, (v, v))$ holds, which is a contradiction. The proof is now complete. ◇

Chestnuts will play an important role in the description of deterministic soliton graphs. The following propositions provide the structural characterization of chestnuts and describe the automata associated with such graphs.

Proposition 6.2.3 *A connected graph G is a chestnut iff it is a bipartite viable soliton graph subject to the following conditions:*

(a) Each external edge of G is constant.

(b) G contains a unique non-mandatory internal elementary component and this component consists of an even-length cycle.

(c) For any family $\mathcal{F}$ of G there exists at most one edge e such that $e : \mathcal{F} \mapsto \mathcal{F}'$ for some family $\mathcal{F}'$.

Proof. *Only if.* Assume that $G = \alpha + \beta_1 + \ldots + \beta_k$ ($k \in N$) is a chestnut such that α is an even length cycle, and for each $i \in [k]$, β_i is a tree having a unique common vertex v_i with α. Then construct a perfect internal matching M with the following conditions:

(i) α is an M-alternating cycle.

(ii) For $i \in [k]$, any external path of β_i terminating in a vertex with degree greater than 2 is a negative M-alternating path.

It is easy to check that a perfect internal matching M with the above properties exists, because by Definition 2.2.12, any two vertices of β_i ($i \in [k]$) with degree greater than 2 are at even distance from each other. The above facts imply that G is a bipartite viable soliton graph, which proves the first statement of the proposition.

As the next step of the proof, we will show that α is the unique M-alternating unit in G. To this end suppose on the contrary that an M-alternating crossing β exists between external vertices w_i and w_j such that $w_i \in V(\beta_i)$ and $w_j \in V(\beta_j)$ for some $i, j \in [k]$. If $i = j$ stood, then according to condition (ii), both $\beta[w_i, u]$ and $\beta[u, w_j]$ would be negative for any vertex $u \in V(\beta)$ with $d(u) > 2$, which is not possible. Therefore $i \neq j$, consequently β can be decomposed as $\beta = \beta[w_i, v_i] + \alpha[v_i, v_j] + \beta[v_j, w_j]$. However, condition (ii) holds for $\beta[w_i, v_i]$ and $\beta[v_j, w_j]$, by which $\alpha[v_i, v_j]$ is a positive alternating path. Hence we obtained a contradiction, as the length of $\alpha[v_i, v_j]$ is necessarily even by the definition of chestnuts.

Summarizing the observations of the previous paragraph, we conclude that α constitutes an elementary component C', and any additional internal elementary component is mandatory. Moreover, each external edge is obviously constant, and by Proposition 4.6.2, any family consists of a unique elementary component. By the above facts (a) and (b) hold with $\mathcal{F}_{C'}$ being the unique minimal element by $\overset{*}{\mapsto}$.

In order to prove (c) suppose by way of contradiction that there exists a mandatory elementary component C'' incident with two distinct edges e_1, e_2 such that $e_1 : \mathcal{F}_{C''} \mapsto \mathcal{F}_1$ and $e_2 : \mathcal{F}_{C''} \mapsto \mathcal{F}_2$ for some families $\mathcal{F}_1$ and $\mathcal{F}_2$. Now let β_i denote the tree containing e_1, let w denote the common endpoint of e_1 and e_2 belonging to C'', and consider a maximal external alternating trail γ going through e_1. It is clear that γ is a v-racket for some $v \in Ext(G)$, and by Corollary 4.5.23, $\gamma_h[w, v_i]$ is a negative alternating path, i.e. w is at an odd distance from v_i. However, the degree of w is clearly greater than 2, thus a contradiction with condition (iv) of Definition 2.2.12 is obtained.

If. Assume that G is a bipartite viable soliton graph with conditions $(a) - (c)$, and let M be a state of G. Since G is bipartite, by Proposition 4.6.2, each family of G consists of a unique elementary component. Since any internal elementary component is loop-free mandatory except the one – denoted C' – consisting of the cycle α of G, it is easy to see that $\mathcal{F}_{C'}$ is the unique minimal element by $\overset{*}{\mapsto}$. Therefore the principal and non-principal vertices of C' alternate along α, from which condition (iii) of Definition 2.2.12 follows. Now consider a vertex $u \in V(G) \setminus V(C')$ with $d(u) > 2$, and analogously to the argument in the 'Only if' part, consider an M-alternating v-racket γ for some $v \in Ext(G)$ which traverses u. It is clear by the family structure described in (c), that u is a principal vertex, consequently Lemma 4.6.7, part (ii) will imply that the suffix of γ_h from u has an even length, as required. Finally, condition (c) –taking into consideration that each family different from $\mathcal{F}_{C'}$ consists of a single mandatory edge – ensures also that $E(G) \setminus E(\alpha)$ determines a forest, by which the proof is complete. ♢

Corollary 6.2.4 *Let G be a deterministic viable connected soliton graph such that G contains an alternating cycle α with respect to some state $M \in S(G)$, each external edge of G is constant, and α is M-accessible from all external vertices. Then G is a chestnut.*

Proof. By Corollary 5.4.10 and by Lemma 6.2.2, α is contained in the unique non-mandatory internal elementary component C' of G with $\mathcal{F}' \overset{*}{\mapsto} \mathcal{F}_{C'}$ for all external families $\mathcal{F}'$. Therefore, conditions (i) and (ii) of Lemma 6.2.2 hold for C', while $(iii) - (v)$ stand for the families of G different from $\mathcal{F}_{C'}$. The above observations imply that G is bipartite and $\mathcal{F}_{C'}$ is the unique minimal element by $\overset{*}{\mapsto}$. Now we obtain the claim by applying Proposition 6.2.3. ♢

Proposition 6.2.5 *Let G be a viable soliton graph with $Ext(G) = \{v_1, \ldots, v_n\}$ $(n \in N)$. Then G is a chestnut iff the soliton automaton $\mathcal{A}(G) = (S(G), X \times X, \delta)$ is given by the followings:*

(a) $S(G) = \{M_1, M_2\}$.

(b) for any $i, j \in [n]$, $k = 1, 2$

$$\delta(M_k, (v_i, v_j)) = \begin{cases} \{M_{3-k}\}, & \text{if } i = j \\ \{M_k\}, & \text{otherwise} \end{cases}$$

Proof. *'If.'* In this case, one can see that each external edge of G is constant, and the mediator alternating network Γ between M_1 and M_2 consists of a single alternating cycle α being $M_1(M_2)$-accessible from all $v_i \in Ext(G)$. Indeed, condition (b) implies that there does no exist alternating crossing in G, which is equivalent to saying that any allowed external edge of G is mandatory. Furthermore, Γ cannot contain an M_1-alternating cycle β different from α, because in that case, by Theorem 5.2.3, $S(M_1, \beta) \in \delta(M_1, (v_i, v_i))$ would hold for some external vertex v_i from which β is M_1-accessible. Finally, combining Theorem 5.2.3 with condition (b), we conclude that α is indeed $M_1(M_2)$-accessible from each external vertex.

By the observations of the previous paragraph we can apply Corollary 6.2.4 for G, by which G is a chestnut, as required.

'Only if.' Assume that G is a chestnut, and let α denote the unique cycle of G. Now by applying Propositions 4.6.2 and 6.2.3 for G, it is clear that any edge not in α is constant, and the family determined by the elementary component C' containing α is the unique minimal element according to $\overset{*}{\mapsto}$. Therefore condition (a) holds with α constituting the mediator alternating network between states M_1 and M_2, and for $i = 1, 2$ with $j \in [n]$, $M_{3-i} \in \delta(M_i, (v_j, v_j))$ stands by Theorem 5.2.3 and Corollary 5.4.10. Moreover, utilizing that G is bipartite, Theorem 5.2.7 implies for $i = 1, 2$ and $j \in [n]$ that $M_i \notin \delta(M_i, (v_j, v_j))$. Summarizing the above facts, we obtain that (b) also holds with respect to $\mathcal{A}(G)$, which makes the proof complete. ♢

Now we are ready to prove the main result of this section concerning the matching-structure of deterministic soliton graphs.

Theorem 6.2.6 *A soliton graph G is deterministic iff for each connected component G_i of G^+, either G_i is a chestnut or G_i does not contain an alternating cycle with respect to any state of G.*

Proof. *'Only if'.* Assume that G is deterministic and let G_1 be a connected component of G^+ which contains an alternating cycle α with respect to some state M of G^+. We will prove that in this case G_1 is a chestnut.

Observe first that the elementary component C containing α must be internal. Indeed, if C was external, then by Claim 4.2.1, α would be $M_{(C)}$-accessible from some external vertex v of C. However, the above situation implies by Lemma 6.2.1 that v is incident with a constant external edge, which is a contradiction, as C is clearly non-mandatory.

Now let C' be an arbitrary external elementary component with $\mathcal{F}_{C'} \overset{*}{\mapsto} \mathcal{F}_C$. Then combining Claim 4.2.1 with Lemma 5.4.8, we obtain that α is M-accessible from some external vertex $v \in V(C')$. Therefore, by Lemma 6.2.1, C' is either degenerate or mandatory. Since C' was arbitrary, we can conclude that each external edge of the subgraph induced by $X = \{v \in V(C') \mid C' \text{ is an elementary component with } \mathcal{F}_{C'} \overset{*}{\mapsto} \mathcal{F}_C \}$ is constant. Consequently, by applying Lemma 6.2.2 together with Corollary 6.2.4, we obtain that $G[X]$ is a chestnut. We will show that $G[X] = G_1$, by which the proof of this direction will be complete.

Suppose on the contrary that $G[X] \neq G_1$. Then there exists an edge e connecting a family $\mathcal{F}$ of $G[X]$ with a family $\mathcal{F}'$ of $G_1 - G[X]$. It is clear because of the construction of $G[X]$ that only $\mathcal{F} \mapsto \mathcal{F}'$ can hold. Now let $\mathcal{F}''$ denote a minimal element by $\overset{*}{\mapsto}$ such that $\mathcal{F}' \overset{*}{\mapsto} \mathcal{F}''$. Since $\mathcal{F}'' \neq \mathcal{F}_C$, we obtain a contradiction with Lemma 6.2.2, part *(ii)*, which proves that $G[X] = G_1$, as expected.

'If'. Assume now that G is not deterministic. Then, for some state M of G, there exist a pair of external vertices $(v, w) \in Ext(G) \times Ext(G)$ and distinct M-transition networks Γ_1, Γ_2 from v to w. It is clear that each alternating unit of $\Gamma_1 \cup \Gamma_2$ belongs to the same connected component G_i of G^+. Moreover, it is also obvious that $|\Gamma_1 \cup \Gamma_2| > 1$, by which G_i is not a chestnut.

Consider now the mediator alternating network Γ between $S(M, \Gamma_1)$ and $S(M, \Gamma_2)$. It is easy to see that Γ consists of alternating cycles belonging to G_i. Therefore we obtained that G_i is not a chestnut, but it contains an alternating cycle. The proof is now complete. ♢

By Theorem 6.2.6 – with the exception of chestnuts – it is clear that any viable internal elementary component of a deterministic soliton graph is mandatory. Therefore it is enough to characterize elementary deterministic soliton graphs. The above fact is expressed by automata products in Theorem 6.2.8, which is preceded by a simple observation.

Claim 6.2.7 *Let G be a connected viable deterministic soliton graph such that G is not a chestnut. Then $\mathcal{A}(G) = \mathcal{A}_e(G)$.*

Proof. Let M be an arbitrary state of G, and δ be the transition function of $\mathcal{A}(G)$. By Theorem 6.2.6, G does not contain M-alternating cycles, consequently $\delta(M, (v, v)) = \{M\}$ holds for any $v \in Ext(G)$, which proves the claim. ♢

Theorem 6.2.8 *Let $\mathcal{D}$ denote the class of automata obtained by a disjoint product of soliton automata $\mathcal{A}(G_1), \ldots, \mathcal{A}(G_n)$ ($n \in N$) such that G_i ($i \in [n]$) is either a chestnut or an elementary graph without alternating cycles. Then the class of deterministic soliton automata coincides with class $\mathcal{D}$ up to strong isomorphism.*

Proof. The following conspicuous observation will be used throughout the proof.

Claim A Let G be a soliton graph and let $G_1, \ldots, G_n$ ($n \in N$) be the connected components of G^+. Then $\mathcal{A}(G)$ is strongly isomorphic with the disjoint product of $\mathcal{A}(G_1) \ldots, \mathcal{A}(G_n)$.

First we prove for any deterministic soliton graph G that $\mathcal{A}(G)$ is strongly isomorphic with a disjoint product of automata having the required properties. By Theorem 6.2.6 and Claim

A we may assume that G is a connected viable soliton graph having no alternating cycles with respect to any state of G. Now apply Theorem 5.4.14 for $\mathcal{A}(G)$ in order to obtain a canonical product $\mathcal{A} = (S, X \times X, \delta)$ from a system $\mathcal{Q}$ of elementary soliton automata $\mathcal{A}(G_i)$ ($i \in [n], n \in N$) to a system $\mathcal{L}$ of full automata $\mathcal{A}_j$ ($j = n+1, \ldots, m$, $m \geq n$) such that $\mathcal{A}(G) \cong_s \mathcal{A}$. The above fact means that there exists a soliton automaton $\mathcal{A}(G')$ such that $\mathcal{A}(G')$ is strongly isomorphic with $\mathcal{A}$ and $\mathcal{A}_e(G)$ is strongly isomorphic with $\mathcal{A}_e(G')$. However, $\mathcal{A}(G) = \mathcal{A}_e(G)$ by Claim 6.2.7, which implies that $\mathcal{A}_e(G')$ is also a deterministic automaton, consequently $\mathcal{A}(G') = \mathcal{A}_e(G')$. Therefore we can conclude that $\mathcal{A}(G)$ and $\mathcal{A}$ are strongly isomorphic.

Now observe by Theorem 6.2.6 that for each $n+1 \leq j \leq m$, $\mathcal{A}_j$ has a single state. It is easy to see that in this situation if we restrict the canonical productto $\mathcal{Q}$, then we obtain the disjoint product $\mathcal{A}''$ of $\mathcal{A}_e(G_1), \ldots \mathcal{A}_e(G_n)$ such that $\mathcal{A}''$ is strongly isomorphic with $\mathcal{A}$. Moreover, for any $1 \leq i \leq n$, G_i is an elementary component of G, consequently it does not contain alternating cycles. Then applying Claim 6.2.7 for all $\mathcal{A}_e(G_i)$ ($i \in [n]$), the proof of this direction can be finished.

Conversely, let $\mathcal{A}$ be the disjoint product of $\mathcal{A}(G_1), \ldots, \mathcal{A}(G_n)$, where G_i ($i \in [n]$) is either a chestnut or an elementary soliton graph without alternating cycles. Construct the soliton graph G from $G_1, \ldots, G_n$ in such a way that the connected components of G are exactly the given graphs. Therefore, by Claim A, $\mathcal{A}$ is strongly isomorphic with $\mathcal{A}(G)$, and G is deterministic by Theorem 6.2.6. The proof is now complete. $\diamondsuit$

As we have seen, Theorem 5.4.14 fully characterizes the internal part of soliton graphs. Therefore, Theorem 6.2.6 gives a motivation for defining a larger class of "deterministic-type" soliton graphs and automata.

Definition 6.2.9 A soliton graph G is called *partially deterministic* if each external elementary component of G is a deterministic soliton graph. Furthermore, soliton automaton $\mathcal{A}(G)$ is *partially deterministic* if G is such.

Theorem 6.2.10 *Let $\mathcal{C}$ denote the class of automata obtained by a canonical product from a system of elementary soliton automata associated with a graph having no alternating cycles to a system of full automata. Then the class of partially deterministic soliton automata coincide with class $\mathcal{C}$ up to soliton isomorphism.*

Proof. It is easy to see that by applying the construction in the proof of Proposition 5.4.15 for any system of partially deterministic soliton automata, the automaton obtained is also partially deterministic. Now the statement follows from the above observation, and from Theorems 5.4.14, 6.2.6. $\diamondsuit$

6.3 Reducing soliton graphs

According to the structural results of Section 6.2, for the analysis of (partially) deterministic soliton automata, we need to describe the structure of elementary soliton graphs without alternating cycles.

Concerning the above problem, it is a natural conjecture that any elementary graph with an even-length cycle also contains an alternating cycle. However, Figure 6.1 provides a simple counterexample, because in any state of the automaton associated with the given graph must contain one of e and f, by which the unique cycle of G cannot be alternating. Nevertheless, applying an extension of the reducing technique introduced in Section 2.4, we will receive a positive answer for our conjecture in Theorem 6.4.1.

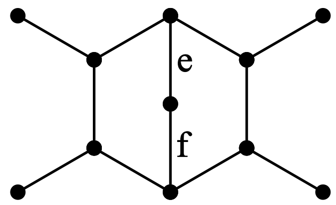

Figure 6.1: Example for deterministic elementary soliton graph with a cycle of even length.

The first reducing operation is contracting a redex in a graph G. Recall from Section 2.4 that if r is a redex in soliton graph G, then G_r denotes the graph obtained by contracting r in G. Furthermore, for any state M of G, M_r denotes the restriction of M to edges in G_r, and for any walk α of G, $trace_r(\alpha)$ denotes the restriction of α to edges in G_r.

Following the same argument as in the the paragraph preceding Proposition 2.4.9, it can be seen that if an alternating unit goes through both focal vertices of a redex r, then it must do so along the center of r. As a consequence we have:

Proposition 6.3.1 *The function $trace_r$ establishes a one-to-one correspondence between alternating units of G and those of G_r. For any M-alternating unit α, $M' = S(M, \alpha)$ holds in G iff $(M')_r = S(M_r, trace_r(\alpha))$ holds in G_r.*

Corollary 6.3.2 *Any edge e of G_r is allowed in G_r iff e is allowed in G.*

Proof. Immediate by Proposition 6.3.1. $\diamondsuit$

The above reduction procedure is extended by another natural simplifying operation on graphs; which is the removal of a loop from around a vertex v if $deg(v) \geq 4$. Such loops will be called *inner*. Let G_v denote the graph obtained from G by removing an inner loop at vertex v. Clearly, if G is a soliton graph, then so is G_v, and the states of G_v are exactly the same as those of G.

Definition 6.3.3 Graph G is called *reduced* if it does not contain a redex or inner loop.

For an arbitrary graph G, contract all redexes and remove all inner loops in an iterative way to obtain a reduced graph $r(G)$. Observe that this reduction procedure has the so called Church-Rosser property (cf. [6]), that is, if G admits two different one-step reductions to graphs G_1 and G_2, then either G_1 is isomorphic to G_2, or G_1 and G_2 can further be reduced to a common graph $G_{1,2}$, actually by the very same reduction steps that were applied for G to obtain G_2 and G_1, respectively. In this context, one reduction step means contracting a redex or removing a single inner loop. As an immediate consequence of the Church-Rosser property, the graph $r(G)$ above is unique up to graph isomorphism.

Making use of the above observations and the first statement of Proposition 2.4.9, we evidently obtain the following.

Proposition 6.3.4 *For any graph G, $r(G)$ is a soliton graph iff G is such.*

Note that, for a soliton graph G, the automata $\mathcal{A}(G)$ and $\mathcal{A}(r(G))$ need not be isomorphic. This follows from the fact that an inner loop around an accessible vertex is the part of an alternating v-loop – for some $v \in Ext(G)$ – inducing a self-transition; which might be not the case if such a loop does not exist. Nevertheless, the following two statements hold.

Proposition 6.3.5 *For any soliton graph G, $\mathcal{A}(G) \cong_s \mathcal{A}(r(G))$.*

Proof. It is enough to prove that one reduction step preserves soliton isomorphism. By Proposition 2.4.9, shrinking a redex r results in a strongly isomorphic soliton automaton $\mathcal{A}(G_r)$, so we can consider the case of eliminating an inner loop. By Theorem 5.2.3, such a reduction does not change the transitions between distinct states, which implies the claim. ◇

Proposition 6.3.6 *For any deterministic soliton graph G, $\mathcal{A}(G)$ and $\mathcal{A}(r(G))$ are strongly isomorphic.*

Proof. We claim that in this case $\mathcal{A}(r(G))$ is also deterministic. For proving the above statement, observe by Proposition 2.4.9, that shrinking a redex r in G results in a deterministic graph G_r. Furthermore, in the light of Theorem 6.2.6 it is also clear that loop elimination does not change determinism either. Therefore, applying the preceding observations in an iterative way, we obtain that $\mathcal{A}(r(G))$ is indeed deterministic.

Now the proof is straightforward by combining the above facts with Propositions 5.4.3 and 6.3.5. ◇

In the rest of this section we prove two important lemmas on reduced soliton graphs.

Lemma 6.3.7 *Let G be an elementary graph containing a cycle. If G is reduced, then there exists an internal edge $h \in E(G)$ such that $G - h$ is still elementary.*

Proof. For a fixed state M of G, let us construct a sequence of subgraphs $G_0, \ldots, G_{n+1}$ of G in the following way. The sequence starts out with the empty graph G_0, and for every $0 \le i \le n$, G_{i+1} is obtained from G_i by adding an M-alternating unit α_i of G which covers at least one edge not already in G_i. The process stops when a connected graph G_{n+1} is reached that covers all vertices of G. Since G is elementary, the process is well-defined. During this process it is possible to cover the external vertices of G first, that is, to select for the unit α_i an appropriate crossing that covers at least one new external vertex, up to an index $i = k$ such that G_{k+1} already covers all external vertices. Moreover, since each internal vertex of G_i is accessible (within G_i) with respect to M, we can assume that each graph G_i, $1 \le i \le k+1$ is a forest.

If G contains edges that are not in G_{n+1}, then we are through, for any, or even all of these edges can be left out from G without losing the elementary property. Assume therefore that $G_{n+1} = G$. Then, clearly, $k < n$. (Remember that G contains a cycle.) On the other hand, observe that the unit α_n could not add a new vertex to G_n. Indeed, such a vertex would necessarily be internal in G, even different from the base vertices, and as such would have a degree greater than two. This contradicts the assumption that $G_{n+1} = G$. We conclude that G_n is not connected, covering all vertices of G though.

Let e be any edge of α_n that connects two different components of G_n. Since all internal vertices of G_n are accessible within their respective connected components, the edge e is allowed in $G_n + e$. Thus, if α_n contains a new edge h which is not a cut edge in G, then $G - h = G_n + \alpha_n - h$ is elementary. Otherwise G_n has a connected component C containing a cycle. Augment C with the new edge(s) of α_n incident with C, considering them as external edges in the resulting graph. This graph satisfies the conditions of the lemma, and has fewer internal edges than G. The proof can now be finished by organizing the above argument as a proper induction on the number of internal edges in G. ◇

A *looping trail* is a trail that consists of a non-empty path with an odd-length cycle attached to either or both of its endpoints. A looping trail is alternating if it alternates on positive and negative edges. Clearly, the odd-length cycles appearing in alternating looping trails are alternating loops.

For any graph G and path (looping trail, odd-length cycle) α in G, *marking* α as an alternating path (respectively, looping trail, odd-length cycle) amounts to specifying a matching M of G which consists of edges along α only, and for which α becomes an M-alternating path (respectively, M-alternating looping trail, M-alternating loop).

Definition 6.3.8 A connected loop-free graph G is a *generalized tree* if it does not contain even-length cycles.

By Definition 6.3.8, the odd-length cycles possibly present in a generalized tree must be pairwise edge-disjoint, which explains the terminology.

Lemma 6.3.9 *Let G be a reduced generalized tree. Then the marking of any path p as an alternating path in G can be extended to a state M_p of G.*

Proof. Again, the proof is an induction argument on the number of internal edges in G. If G has no internal edges, then the statement is trivial. Let G therefore have at least one internal edge, and assume that the statement holds for all reduced generalized trees having fewer internal edges than G. If G has an internal cut edge e, then cut G along e to obtain two generalized trees G_1 and G_2. In both G_1 and G_2, e appears as an external edge, ensuring that these graphs remain reduced. The statement of the lemma then follows easily from applying the induction hypothesis on G_1 and G_2 with suitable alternating paths p_1 and p_2.

If G does not have an internal cut edge, then every internal vertex of G, except possibly the base ones, is part of at least two odd-length cycles, and even the base vertices have a degree at least 3. Consequently, an arbitrary internal edge can be left out from G, still preserving it as a reduced generalized tree. Since p is a path, the edge e to be left out can be chosen from outside p. Now the statement follows directly from the induction hypothesis by taking $e \notin M_p$. $\diamondsuit$

Corollary 6.3.10 *Lemma 6.3.9 holds for all odd-length cycles and looping trails α as well.*

Proof. If α starts (ends) with an odd-length cycle, then delete the first (respectively, last) edge of α to obtain a path p. The statement now follows from marking p in an appropriate alternating way, and applying Lemma 6.3.9 to obtain a suitable state $M_\alpha = M_p$. $\diamondsuit$

6.4 Characterizing deterministic and partially deterministic soliton automata

Now we are ready to prove the main results of the chapter. The key to this results is Theorem 6.4.1 below. The underlying idea of the proof of this theorem is a simple induction argument, which is highlighted by the second paragraph of the proof. Yet, the complete proof is technically challenging, as one must deal with a large number of cases and subcases.

Theorem 6.4.1 *Let G be a reduced elementary soliton graph. If G contains an even-length cycle, then it also has an alternating cycle with respect to some state of G.*

Proof. Induction on the number of internal edges of G. If G has no internal edges, then we have nothing to prove. Let G have at least one internal edge, and assume that the statement holds for all appropriate graphs having fewer internal edges than G. If G has a pair $\{h, g\}$ of parallel edges, then either both, or none of these edges is allowed. If they are allowed, then they form an alternating cycle with respect to every state M in which one of them is present. If

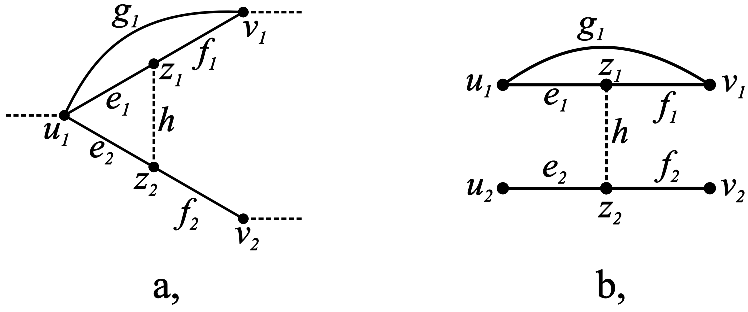

Figure 6.2: The proof of Theorem 6.4.1

they are forbidden, then the graph $G - h$ is still elementary and reduced, otherwise some edge adjacent to h and g would be mandatory in G. Now the statement follows from the induction hypothesis.

We can therefore assume that G does not contain parallel edges. Let G, nevertheless, contain an even-length cycle. By Lemma 6.3.7 there exists an internal edge $h \in E(G)$ such that $G - h$ is still elementary. Suppose first that $G - h$ is reduced. If it also contains an even-length cycle, then by the induction hypothesis we are through. If not, then $G - h$ is a reduced generalized tree, and G has an even-length cycle α containing h. Now the statement is obtained by applying Lemma 6.3.9 on $G - h$, marking $\alpha - h$ as a positive alternating path in that graph.

Suppose now that $G - h$ is not reduced, and let z_1, z_2 denote the two endpoints of h with z_1 being the center of an appropriate redex r in $G - h$. Since h is the only edge in G connecting z_1 and z_2, the focal vertices u_1, v_1 of the redex r are both different from z_2. We organize the rest of the proof according to whether both, only one, or neither of u_1 and v_1 is adjacent to z_2 in G. Let G' denote the graph obtained from G by contracting the redex r to a sink vertex s and removing a possible loop around s caused by the presence of the edge $g_1 = (u_1, v_1)$ in G. By Corollary 6.3.2, G' is elementary.

Case 1: both u_1 and v_1 are adjacent to z_2.

In this case G' will contain a pair of parallel edges connecting z_2 with the sink s. Graph G' is also reduced, otherwise it would not be elementary. Now the statement follows from the induction hypothesis and Proposition 6.3.1.

Case 2: only one of u_1 and v_1, say u_1, is adjacent to z_2.

To handle this case, we need to further break it down into two subcases.

Subcase 2a: the edge $g_1 = (u_1, v_1)$ is present in G, as shown in Fig. 6.2a.

If the edge $e_2 = (u_1, z_2)$ is allowed in $G - h$, then let M be any state of G for which $e_2 \in M$. Clearly, the edges e_2, h, $f_1 = (z_1, v_1)$, and g_1 form an M-alternating cycle. If e_2 is forbidden in $G - h$, then G' must be reduced. For, if G' was not reduced, then either z_2 or the sink s would become the center of a new redex r' in G'. Either way, e_2 would be one of the focal edges of r', implying that the other focal edge is mandatory; a contradiction. The graph G' cannot be a generalized tree, because in that case Lemma 6.3.9 would force e_2 to be allowed. We conclude that G' is reduced and contains an even-length cycle. Now the statement follows again from the induction hypothesis and Proposition 6.3.1.

Subcase 2b: the edge $g_1 = (u_1, v_1)$ is not present in G.

First assume that $deg(z_2) \geq 3$. Then G' is again reduced. If it also contains an even-length cycle, then we are through. Otherwise G' is a generalized tree, and G has an even-length cycle

α passing through h. Concentrate on the trace α' of α in G'. It will consist of a possible odd-length cycle around the sink s, and a path p connecting s with z_2. (See again Fig. 6.2a.) If the odd-length cycle is present in α', then the length of p is odd, too, because α' contains two less edges than α. If not, then the length of p is even. In this case, however, α covers only one of u_1 and v_1. Should either of them be missed by α, there exists an edge $t \in E(G)$ incident with that vertex such that $t \notin \{e_1, f_1, e_2\}$. (Recall that $g_1 = (u_1, v_1)$ is now not present in G.) Moreover, t is not incident with any vertex on α, otherwise t would give rise to an even-length cycle in G'.

Mark the edges of α' in an alternating way, so that the endpoint z_2 of α' be positive. Furthermore, if $\alpha' = p$, then add the edge t specified above with a positive sign to the marking. By Corollary 6.3.10, this marking can be extended to a state M' of G' by which α' is an M'-alternating path or looping trail, positive at its z_2 end. Moreover, if $\alpha' = p$, then $t \in M'$. Finally, reconstruct the cycle α from α', and observe that α is M-alternating in G with respect to the unfolding M of M' to $G - h$, and further to G by taking $h \notin M$.

Now assume that $deg(z_2) = 2$, so that G has a redex around both z_1 and z_2. By symmetry, we can assume that $g_2 = (u_1, v_2)$ is not present in G either. Then the graph G'' obtained from G by contracting its two redexes and removing a possible loop (caused by the edge (v_1, v_2)) around their common sink s is reduced. Again, if G'' contains an even-length cycle, then we are done. If not, then G'' is a generalized tree, and we proceed as in the second last paragraph. The trace α' of the even-length cycle α is now either empty, when α consists of the four edges f_1, h, f_2, (v_1, v_2), or it is a single odd-length cycle, when (v_1, v_2) is not present in G. Indeed, if $(v_1, v_2) \notin E(G)$, then α' has three less edges than α, so that if α' consisted of two cycles, then one of these would be even-length. Either way, α misses one of the vertices u_1, v_1, v_2. No matter which one of them is missed, there exists an edge $t \in E(G)$ incident with that vertex, such that the other endpoint of t is not on α either.

Mark the edge t positive in G'', and continue the marking on α' in an alternating way if α' is not empty. Using Corollary 6.3.10, extend this marking to a state M'' of G''. Reconstruct α from α', and observe that α becomes an M-alternating cycle with respect to the unfolding M of M'' to $G - h$, and on to G by taking $h \notin M$.

Case 3: neither u_1 nor v_1 is adjacent to z_2.

Subcase 3a: the edge $g_1 = (v_1, v_2)$ is present in G, as depicted in Fig. 6.2b.

Connect z_2 with the sink s by a new edge h' in G'. Then $G' + h'$ is reduced. Assume that $G' + h'$ contains an even-length cycle. Then, by the induction hypothesis, $G' + h'$ has an alternating cycle α with respect to some state M'. If α does not go through h' and $h' \notin M'$, then the unfolding of M' from G' to G preserves α as an alternating cycle. Otherwise we can assume that $h' \in M'$. Consider the matching $M = M' - h' + h + g_1$ in G. Clearly, M is a state, and if α goes through h', then the positive edge h' can be substituted in α by the 3-length positive M-alternating path in G consisting of the edges h, g_1, and one of the focal edges of r.

Subcase 3b: the edge $g_1 = (u_1, v_1)$ is not present in G.

By symmetry, we can again assume that if $deg(z_2) = 2$, then $g_2 = (u_2, v_2)$ is not present in G either. Contract the one or two redexes in $G - h$ to obtain a reduced elementary graph G''. If G'' still contains an even-length cycle, then we are finished. Otherwise G'' is a generalized tree, and G has an even-length cycle α going through h. As in Case 2, concentrate on the trace α' of α in G''. It turns out that α' consists of a path p with a possible odd-length cycle attached to either or both of its endpoints. A cycle is present at an endpoint of p iff that endpoint is a sink s_i $(i = 1, 2)$ in G'' and both vertices u_i, v_i are on α. Furthermore, the number of edges in α' is 3 or 2 less than in α, depending on whether the sink s_2 exists or not.

Assume that the sink s_i exists, but only one of u_i, v_i (say v_i) is on α. Then u_i is either base or $deg(u_i) \geq 3$. Either way, there exists an edge t_i incident with u_i but not incident with any vertex on α. (Notice that the presence of two edges connecting u_i with vertices on α would inevitably lead to an even-length cycle in G''.) Moreover, if both indices $i = 1, 2$ are considered in this argument, then t_1 and t_2 can be chosen in such a way that they are not identical, and not even adjacent. (Again, otherwise G'' would contain an even-length cycle.)

Synthesizing the above, we proceed with marking α' as follows.

— Mark the odd-length cycles present in α' as alternating loops.

— If either endpoint of p is a sink s_i, but an odd-length cycle is not present at s_i in α', then mark the edge t_i positive.

— Mark the path p in an alternating way, so that the marking be positive at either end iff the corresponding endpoint is not a sink or has an odd-length cycle attached to it in α'.

It is easy to see that the marking procedure above is consistent, and it results in an alternating path or looping trail covering the edges in $\alpha' \cup \{t_1, t_2\}$. Extend this marking to a state M'' of G'' using Corollary 6.3.10, and observe that α becomes an M-alternating cycle with respect to the unfolding M of M'' to G. The proof of Theorem 6.4.1 is now complete. ♢

Corollary 6.4.2 *For any graph G, if $r(G)$ is a generalized tree, then G is a deterministic soliton graph. Conversely, if G is a non-mandatory deterministic elementary soliton graph, then $r(G)$ is a generalized tree.*

Proof. Clearly, G is a soliton graph iff $r(G)$ is such. By Proposition 6.3.1, if $r(G)$ is a generalized tree, then G does not contain alternating cycles with respect to any of its states. Theorem 6.2.6 then implies that G is deterministic. Conversely, if G is a deterministic elementary soliton graph, then so is $r(G)$, containing no alternating cycles with respect to any of its states. (See again Theorem 6.2.6 and Corollary 6.3.2.) Thus, by Theorem 6.4.1, $r(G)$ is a generalized tree. ♢

Corollary 6.4.3 *A non-mandatory elementary soliton graph is deterministic iff it reduces to a generalized tree.*

By the above results we can give the characterization of the classes of deterministic and partially deterministic soliton automata. To this end let G_m denote the mandatory elementary soliton graph with a unique loop around its internal vertex, and let $\mathcal{T}$ denote the class of soliton automata $\mathcal{A}(G)$ such that either $G = G_m$ holds or G is a reduced generalized tree.

Theorem 6.4.4 *The class of deterministic elementary soliton automata coincide with the class $\mathcal{T}$ up to strong isomorphism.*

Proof. It is clear that for any mandatory elementary soliton graph G, $\mathcal{A}(G)$ is a full automaton with a unique state, which implies that $\mathcal{A}(G)$ is deterministic and strongly isomorphic with $\mathcal{A}(G_m)$. Therefore the theorem is a straightforward consequence of Proposition 6.3.6 and Corollary 6.4.3. ♢

Corollary 6.4.5 *Let $\mathcal{C}$ denote the class of automata obtained by a canonical product from a system of soliton automata in $\mathcal{T}$ to a system of full automata. Then the class of partially deterministic soliton automata coincide with class $\mathcal{C}$ up to soliton isomorphism.*

Proof. Immediate by Theorems 6.2.6, 6.2.10 and 6.4.4. ♢

Graph G is called a *baby chestnut* if it consists of two parallel edges connecting a pair of vertices (v_1, v_2), and a number of external edges, each of which connects an external vertex with v_1. It is clear that there exists a unique baby chestnut – denoted by B_k – with k external vertices for any $k \in N$.

Now let $\mathcal{D}$ denote the class of soliton automata $\mathcal{A}(G)$ such that either $\mathcal{A}(G)$ belongs to $\mathcal{T}$ or $G = B_k$ for some $k \in N$. Then the class of deterministic soliton automata is characterized in the following way.

Corollary 6.4.6 *The class of deterministic soliton automata and the class of automata obtained by a disjoint product of soliton automata in $\mathcal{D}$ coincide up to strong isomorphism.*

Proof. By Proposition 6.2.5, any chestnut with k external vertices ($k \in N$) is strongly isomorphic with $\mathcal{A}(B_k)$. Then the statement is the immediate consequence of Theorems 6.2.6, 6.2.8 and 6.4.4. $\diamondsuit$

Finally we present the algorithmic consequences of our results. As was mentioned earlier, by definition we do not receive an efficient way for checking determinism of a given soliton graph. Indeed, the graph might have exponential number of states, which results in an exponential time algorithm for ACP. The result of Section 6.2 was not suitable for a polynomial time algorithm either, because for testing the existence of an alternating cycle, we should list all the states, which causes the same problem, like above. However, combining Theorem 6.4.3 with our algorithmic results concerning elementary decomposition, we obtain a straightforward way to test if a soliton graph is deterministic. For this, let G be an arbitrary soliton graph, let $n = |V(G)|$, $m = |E(G)|$ and consider the following algorithm.

0 Construct the viable part G^+ of G by the algorithm of Section 4.7. Determine the connected components of G^+. For each connected component G_i of G^+ execute the following steps.

1 Using the method of Theorem 5.5.9, determine the elementary decomposition of G_i. If G_i contains a non-mandatory internal elementary component, then with the help of Lemma 6.2.3 test if G_i is a chestnut. In the case of a negative answer stop the algorithm with the output that G is not deterministic.

2 Apply the reduction procedure for each external elementary component C_{ij} of G_i, and test the existence of an even-length cycle in $r(C_{ij})$. If one of the reduced external elementary components is not a generalized tree, then stop the algorithm with the output that G is not deterministic

If our algorithm did not stop with a negative answer, then we conclude that G is deterministic. The correctness of our algorithm follows from Theorem 6.2.6 and Corollary 6.4.3. Step 0 runs in $\mathcal{O}(m)$ time according to the result of Section 4.7 and the well-known algorithms for isolating the connected components(see e.g. [4]). The running time of the algorithm suggested by Theorem 5.5.9 is $\mathcal{O}(n^3)$, while testing if a graph is a chestnut can be implemented in $\mathcal{O}(m)$ time in a straightforward way. The reduction procedure takes $\mathcal{O}(n \cdot m)$ time, since once a shrinking action occurs, the change in the structure concerns at most n vertices. Finally, for the testing the existence of an even-length cycle we can apply a depth-first search ([94]), which has $\mathcal{O}(m)$ running time. Therefore we obtained the following closing result.

Theorem 6.4.7 *It can be checked in $\mathcal{O}(n^3)$ time if a soliton graph is deterministic.*

Conclusions and further research topics

In this thesis we have given a detailed structural analysis of soliton graphs and soliton automata on the basis of graph matchings. First we have shown that our concept is equivalent to the original definition ([34]) based on weighted underlying graphs, which has been not proved earlier.

Then we have proved two Tutte type theorems for maximal splitters and maximal inaccessible splitters. These results have shown that maximal splitters serve as extreme sets with respect to the counterpart of Tutte theorem for perfect internal matchings, but only the maximal inaccessible splitters are "real" barriers.

We have given a generalization of the canonical partition originally introduced for elementary graphs only, and with the help of this new concept we have built up a structure among the elementary components, by which the viable components could be grouped into disjoint families. We have shown that the families themselves are arranged in a partial order $\overset{*}{\mapsto}$ according to the order they can be covered by external alternating paths. Moreover, we have characterized the subgraphs determined by the families, which result led to a linear-time algorithm for isolating the families.

By giving a characterization of soliton walks with the help of alternating networks and alternating units we presented an algorithm solving the Automata Construction Problem. Then we have characterized soliton automata with a single input as being full or semi-full automata. Making use of the structure theory with respect to elementary graphs, we have proved that the class of soliton automata coincide with the class of automata obtained by a canonical product from a system of elementary soliton automata to a system of full automata up to soliton isomorphism. We have defined the Automaton Description Problem (ADP) for soliton graphs, and worked out the Elementary Structure Encoding of soliton graphs which is equivalent to the original graph concerning ADP. We have given an algorithm which constructs the Elementary Structure Code of a graph in polynomial time iff the state complexity of all the internal elementary components can be determined in polynomial time.

Finally, we have given a characterization of deterministic soliton automata and partially deterministic soliton automata by disjoint and canonical products of generalized trees, chestnuts and mandatory elementary graphs. As an application, we have presented an algorithm which decides in $\mathcal{O}(n^3)$ time if a graph is deterministic.

We believe that the above results will have a real impact on the design and verification of soliton circuits, as outlined by some algorithms obtained as consequences of our structure theory. Nevertheless, the further improvement of these results towards practical applications needs consultation with the engineering profession. Moreover, our results can induce several research problems for future work in graph theory and in automata theory, as well. A couple of them are presented below.

- We have reduced the analysis of soliton automata to elementary soliton automata. Future research has to concentrate on these automata concerning their theoretical characterization, and practical issues as well. Related to this problem, a central question is to find classes of elementary automata for which ADP can be efficiently solved.

- We have faced the problem several times in the thesis that if an algorithm for closed graphs is extended to open graphs by the closure operation, then the complexity of the algorithm is increased because of the marginal edges. However, by a careful implementation, all of these algorithms can be modified in such a way that the original complexity is preserved. Nevertheless, it would be important to find a general method by which this reduction could be applied automatically.
- The above observation shows that the algorithm testing if a graph is deterministic can be implemented in $\mathcal{O}(n \cdot m)$ time too. However, we believe that there exists a linear time algorithm for the above problem. The first results in reaching the above goal are presented in [20].
- We have given an algorithm with a complexity $\mathcal{O}(n \cdot m)$ for searching a double v-racket. However, Corollary 5.2.5 and Proposition 5.2.12 together show that if double v-rackets couldbe identified also in linear time, then the complexity of ACP would be improved. Actually [22] suggests such a method, but the analysis of this algorithm is not complete yet.
- Considering the computational power of soliton circuit it is also an important question to find those elements by which all systems can be simulated. From automata theoretic point of view it means that continuing the research of [58], the isomorphically complete systems of deterministic and nondeterministic automata are to be described. Preliminary results are summarized in [80].
- Having characterized the structure of deterministic soliton graphs it is also a natural idea to investigate the transition monoids of deterministic soliton automata. It is clear that for this research only reduced generalized trees are to be considered. Generalized trees also have a recursive structure which makes it possible to study them in an analogous fashion to the techniques used in former works ([34], [35], [36]).
- Graph-expressions are also powerful enough to describe graph-structures ([23]). Moreover, for the future research, a formal tool enabling to generate soliton graphs with given properties (e.g. deterministic soliton graphs) is also a demand. The above facts motivate to define context-free grammars generating graph-expressions for soliton graphs. In this topic we also have some preliminary results ([16], ([13], [16]).
- It can be proved (cf. [79]) that several classical results from matching theory, such as Gallai-Edmonds Structure Theorem ([53], [54], [40]), Dulmage-Mendhelson decomposition for bipartite graphs ([38], [39]), the Cathedral construction for saturated graphs ([85]), Kotzig's lemma ([75]), can be generalized for S-matchings as an easy consequence of our structure theory. Although the first steps have been made ([21], [79]), but further research in this line can be quite fruitful.
- The perfect path-matching problem ([32]) as a common generalization of matchings and matroid intersections ([41]) is a fascinating subject of current research in graph theory, which is defined as follows. Let G be an undirected graph and $T_1, T_2 \subseteq V(G)$ be disjoint vertex sets with $|T_1| = |T_2|$, called terminal sets. Decide if there exists a *perfect*

path-matching, i.e. a subgraph G' of G such that $V(G') = V(G)$, and each connected component of G is either a path connecting vertices in distinct terminal sets or a single edge. It is easy to notice that, with the parameter $S = V(G)\setminus(T_1 \cup T_2)$, a perfect path-matching is a special perfect S-matching. However, so far only the most fundamental structure theorems have been proved for path-matchings: the counterpart of the Tutte-Berge formula ([45]) and that of the Gallai-Edmonds structure theorem ([90]). Furthermore, though the problem is in P as it was shown in [32] by the ellipsoid method ([64]), but no practically efficient algorithm has been developed so far. The only combinatorial algorithm in the topic ([91]) is quite complicated to understand and no efficient implementation is known. Our structure theory can reduce the problem size to the level of elementary graphs with respect to S–matchings. A more sophisticated description of elementary graphs might result in a better understanding of the structure of the path-matchings, which can also have algorithmic consequences.

- Since finding the minimal $k-$(edge-)connected spanning subgraph is NP-hard (cf.[33], [46]), current research tends to develop approximation algorithms for the problem. Heuristics based on matching theory is one of the most popular approaches (see e.g. [29],[30]). By the external network problem ([67], we can obtain an extension of the problem for open graphs. Naturally, just like perfect matchings play central role for the classical k-connection problem, the structural study of perfect internal matchings may result in efficient heuristics for external networks. Moreover, a possible approximation method for the original k-connected subgraph problem might be the decomposition of the graph into smaller subgraphs such that each subgraph would be considered as an external network (designating those vertices as external vertices by which the subgraph is connected to the remaining part of the graph). Then we would search suboptimum in each subgraph and these smaller spanning subgraphs could be connected in an appropriate way as a solution. Of course, for such an approach no better theoretical bound can be given as approximation ratio, but in practice it might produce good performance.

- The first natural generalization of matchings was the f-factor problem ([98]). An f-factor of a graph is a spanning subgraph having a prescribed degree $f(v)$ at each vertex v. If we drop the above degree-prescription for designated vertices, called terminal vertices, then as a generalization of S-matchings, we obtain an open f-factor problem. Such an open f-factor problem is suitable to model all open conjugated systems, i.e. systems in which a molecule corresponding to an internal vertex might join its surroundings by several double bonds. Alternating walks between terminal vertices can be defined in an analogous fashion to soliton walk. Therefore automata associated with graphs having some prescribed open f-factor can be naturally defined. A further development of the structure theory for f-factors in [84] might be the basis for the analysis of these automata.

Acknowledgements

I am grateful to Miklós Bartha for his expert guidance and valuable suggestions during the course of this work. I am especially grateful to him for making it possible to visit Memorial University of Newfoundland and providing opportunity for taking part in a fruitful research area.

Summary

This thesis deals with soliton automata, which is the mathematical model of certain possible molecular electronic devices called soliton circuits. In a soliton circuit all the chemical components are interconnected by single strands of polyacetylene. Solitons through these components will cause changes in their state, in much the same manner, as a soliton changes the state of polyacetylene.

The model of soliton automata was introduced by J. Dassow and H. Jürgensen in 1990 in order to capture the logical aspects of the "valve" effect by which soliton switches and soliton circuits might operate. The underlying object of a soliton automaton is the so-called soliton graph representing the topological structure of the corresponding molecule-network. In this model atoms (or groups of atoms) are represented by vertices and chemical bonds correspond to edges. The vertices with degree 1 are designated as external vertices, while a vertex with degree greater than one is called internal. External vertices correspond to the marginal parts of the system, which parts serve as electron donors or acceptors for the remaining part of the molecule-network. The internal vertices correspond to an atom (or group of atoms) with the property that among its neighbors there exists a unique one to which it is connected by a double bond. The above property is described by a perfect internal matchings, matchings which cover all the internal vertices. Therefore a soliton graph necessarily possesses perfect internal matchings, also referred to as states, representing the states of the system.

A soliton automaton is defined by its underlying soliton graph G in the following way. The states of the automaton are the states of G, the input alphabet consists of the ordered pair of external vertices, and the state transitions are induced by making alternating walks between the external vertices given by the input. The effect of such a walk is that the status of each edge is exchanged dynamically step by step while making a walk, and by the time the walk is finished a new state is reached.

The analysis of soliton automata is a complex task, earlier works could describe only a few special cases ([34], [35], [36], [37] and [58]. However, no detailed theory has been developed for the description of the underlying topological structure of these automata, which explains the lack of more general results on soliton automata.

This thesis is motivated by the above recognition and its goal is to provide a detailed structural description of soliton graphs and soliton automata. The algorithmic consequences of these results are also outlined.

The first chapter presents the common preparatory notion, notation and terminology.

In Chapter 2, we review the preliminary results concerning soliton automata. The original definition was based on a weighted graph model. Here we present a model on the principle of graph matchings, and proves its equivalence to the original concept.

Then we present our results in detail as follows.

In Chapter 3, first we introduce the concept of splitters in soliton graphs: a set of internal vertices of graph G is called splitter if connecting any two of its elements by an edge e, the edge will be forbidden, i.e. not contained in any state, in the resulted graph $G + e$. Moreover,

the concept of factor-critical graphs is also generalized: A connected graph G is factor-critical if for every internal vertex v, G has a matching covering every internal vertex but v.

Then making use of the new concepts we prove two Tutte type theorems on splitters. The first theorem provides a characterization of maximal splitters, while the second one shows an important property of maximal inaccessible splitters (a splitter is inaccessible if it does not contain any vertex accessible by an external alternating path).

Theorem. *Let X be a non-empty set X of internal vertices of a soliton graph G, and let $c_{\text{in}}(G, X)$ denote the number of connected components of $G - X$ containing internal vertices only. Then the following two statements are equivalent.*

(i) The set X is a maximal splitter.

(ii) Each non-degenerate connected component (consisting of a single external vertex) of $G - X$ is factor-critical such that

(iia) $|X| = c_{\text{in}}(G, X) + 1$, or

(iib) $|X| = c_{\text{in}}(G, X)$ with every external component of $G - X$ being degenerate.

Furthermore, condition (iib) holds in (ii) above if and only if X is inaccessible.

Theorem. *An open graph G is a soliton graph if and only if $c^{\text{o}}_{\text{in}}(G, X) \leq |X|$ for all $X \subseteq Int(G)$, where $c^{\text{o}}_{\text{in}}(G, X)$ denotes the number of odd connected components of $G - X$ containing internal vertices only. Equality may hold for some non-empty X only if not all external connected components of G are factor-critical. In this case, the equation is guaranteed by any maximal inaccessible splitter X.*

In Chapter 4, after proving some technical lemmas, we generalize the canonical partition of elementary graphs (allowed edges of which form a connected spanning subgraph) for all graphs having a perfect internal matching. This partition is given by the sets obtained as the restriction of a splitter to an elementary component. Based on this partition the elementary components containing viable edges (edges traversed by an external alternating trail) are given a structure reflecting the order in which they can be reached by external alternating paths. The observations of this structure is summarized below.

Theorem. *The viable elementary components can be grouped into disjoint families such that the following conditions hold.*

(i) Any family contains at most one external elementary component, which families are called external.

(ii) There exists a unique canonical class P, called principal canonical class of $\mathcal{F}$, in any internal family(family not containing external elementary component) $\mathcal{F}$ such that any external alternating path leading to a member of the family must reach P first. The elementary component containing P is called the root of the family.

(ii) There exists a partial order $\overset{*}{\mapsto}$ among the families reflecting the order by which external alternating paths reach the families. The maximal elements are the families containing an external vertex.

(iv) An edge incident with a vertex in the viable part of a soliton graph is impervious iff both of its endpoints belong to the principal canonical class of some one-way internal elementary component.

Then we characterize families in terms of splitters and factor-critical graphs. This characterization leads to a linear time algorithm isolating the viable families.

In Chapter 5, we define the Automaton Construction Problem (ACP) for soliton graphs and give a matching-theoretic characterization of soliton transitions leading an algorithmic solution of ACP.

Theorem. *Let G be a soliton graph and $k = |S(G^+)|$. Then ACP can be solved in $\mathcal{O}(k^2 \cdot n \cdot m)$ time.*

Then, generalizing a result of Dassow and Jürgensen, we characterize soliton automata with a single external vertex by showing that these automata are either full (there is a transition between any two states) or semi-full (there is transition between any two distinct states, but there is no self-transition). This result play an important role in the decomposition result, which is based on the so-called canonical products. A canonical product is a special type of α_0^ε-product such that the automata on the second level are connected to the soliton automata on the first level, through their canonical classes, according to a canonical dependency, which is simply a mapping from the set of automata on the second level to the power set of canonical classes of soliton automata on the first level. State transition is induced in an automaton on the second level according to its "accessibility" from the first component of the input pair through a canonical class determined by the canonical dependency. The main result is described with the help of the concept of soliton isomorphism, by which we mean a strong isomorphism between the extensions of the given automata in which the empty soliton walk is also allowed.

Theorem. *The class of soliton automata and the class $\mathcal{S}$ of automata obtained by a canonical product from a system of soliton automata to a system of full automata coincide up to soliton isomorphism.*

Finally we investigate the Automaton Description Problem (ADP) for soliton graphs both from descriptional and computational point of view. We work out the so-called Elementary Structure Encoding by which each soliton automaton has a code with better descriptional complexity for ADP. Combining this result with the algorithmic solution of ACP, we obtain the following sufficient condition for a polynomial time method for ADP.

Theorem. *Let G be a soliton graph such that each of its external elementary components has a polynomial number of states and the state complexity of each internal elementary component of G can be determined in polynomial time. Then ADP can be solved in polynomial time for G.*

In Chapter 6, first we give a matching-theoretic characterization of deterministic soliton graphs, i.e. we prove that a graph is deterministic iff each connected component of its viable part is either a chestnut (a graph consisting of an even cycle and a few trees such that any two vertices with degree greater than 2 are at even distance from each other) or it does not contain an alternating cycle of even length. The concept of partially deterministic soliton automata is also defined, as automata associated with a graph such that each of its external elementary components is deterministic. In order to obtain a matching independent characterization of deterministic and partially deterministic automata, we introduce a reduction method for soliton graphs, which preserves isomorphism. We prove that an elementary graph is deterministic iff it can be reduced to a graph without even-length cycle, called a generalized tree. This result leads then a to more sophisticated description of the class of partially deterministic and deterministic soliton automata by canonical and disjoint products (quasi-direct canonical products) of reduced generalized trees and that of baby chestnuts (chestnuts consisting of two parallel edges and a number of external edges having their internal endpoints in common):

Theorem. *Let $\mathcal{T}$ denote the class of soliton automata associated with either a reduced generalized tree or a mandatory elementary graph. Furthermore, let $\mathcal{D}$ denote the class of soliton automata $\mathcal{A}(G)$ such that either $\mathcal{A}(G)$ belongs to $\mathcal{T}$ or G is a baby chestnut. Then the followings hold.*

(i) The class of partially deterministic soliton automata and the class of automata obtained by a canonical product from a system of soliton automata in $\mathcal{T}$ to a system of full automata coincide up to soliton isomorphism.

(ii) The class of deterministic soliton automata and the class of automata obtained by a disjoint product of soliton automata in $\mathcal{D}$ coincide up to strong isomorphism.

The above characterization of the graph structure of reduced deterministic elementary graphs results in a $\mathcal{O}(n^3)$ time algorithm deciding if a graph is deterministic. This algorithm consists of three methods: the construction of the elementary decomposition of the given soliton graph, the reduction procedure for the external elementary components, and a method testing the existence of a cycle of even length in the reduced external elementary components.

Finally, the closing chapter of the thesis is devoted to discussing some open problems.

This thesis is strongly based on the papers [11], [15], [17], [18], [19], and [78].

Összefoglaló (Summary in Hungarian)

Ezen disszertáció témája a szoliton automata vizsgálata, amely egy lehetséges bioelektronikai kapcsolóhálózat, az úgynevezett szoliton áramkör matematikai modellje. Egy szoliton áramkörben a kémiai komponensek, melyek kapcsolóelemként szolgálnak, szénhidrogén-molekulaláncokkal lennének összekötve, amelyen keresztül az elektronok úgynevezett ”szoliton hullámok” formájaban áramolnának. Egy ilyen hullám a rendszerben állapotátváltozást idéz elő amint az útja során a molekulaláncban felcseréli az egyes és kettős kötéseket.

A szoliton automata modelljét Jürgen Dassow és Helmut Jürgensen definiálta 1990-ben, hogy a szoliton hullámok kapcsolóelemként való használhatósága logikai szempontból lehetővé váljék. A szoliton automata alapobjektuma az úgynevezett szoliton gráf, amely a megfelelö molekula-lánc topológiai leírására szolgál. Ebben a modellben a gráf csúcspontjai az atomoknak (vagy az atomok egy csoportjának) felelnek meg, míg a kémiai kötéseket a gráf élei reprezentálják. Azon csúcspontokat melyek fokszáma 1 úgynevezett külső csúcsokként különböztetjük meg, míg azon csúcsok, melyek fokszáma legalább 2, belső csúcsoknak nevezzük. A külső csúcsok a rendszer interfészeként szolgálnak, melyeken keresztül az elektronok a molekula-hálózat belső struktúráját elérhetik. A belső csúcsok olyan atomoknak (vagy atomok egy csoportjának) felelnek meg, melyek pontosan egy szomszédjukhoz kapcsolódnak kettős kötéssel. Ezt a tulajdonságot a gráfmodellben a teljes belső párosításokkal írhatjuk le, azaz olyan párosításokkal, melyek minden belső csúcsot lefednek. A fentiekből adódóan egy szoliton gráf mindig rendelkezik egy teljes belső párosítással, melyeket a gráf állapotaink is szoktunk nevezni utalva a megfelelő molekulalánc álapotaira.

A szoliton automata az alapobjektumaként szolgáló szoliton gráf segítségével definiálható. Egy adott G szoliton gráf esetén, a G-hez rendelt $\mathcal{A}(G)$ szoliton automata állapothalmazát a G állapotai alkotják, az input ábécé a külső csúcsok rendezett párjaiból áll, míg az átmeneteket az inputként adott külső csúcsok közötti alternáló séták realizálják. Egy ilyen alternáló séta során az érintett élek státuszai az adott teljes belső párosításra vonatkoztatva dinamikusan változnak, és bár a séta nem minden lépése eredményez egy állapotot (általában még egy párosítást sem), az input második komponenseként szolgáló külső csúcsot elérve a gráf (és így az automata) egy újabb állapotához jutunk.

A szoliton automaták analízise komplex feladatnak bizonyult, a témában született első dolgozatok csak bizonyos speciális determinisztikus esetek vizsgálatára szorítkoztak ([34], [35], [36], [37] és [58]), azonban egy olyan elmélet kifejlesztése továbbra is váratott magára, amely az alapobjektumként szolgáló szoliton gráfok struktúrális leírása segítségével alapot szolgáltatott volna a szoliton automaták további analíziséhez.

Ezen tézis motivációját a fenti felismerés adta és célként fogalmazódott meg, hogy a szoliton gráfok és ennek alapján a szoliton automaták egy részletes struktúrális leírását adjuk meg. A disszertációban szereplő elméleti eredmények algoritmikus következményei pedig a szoliton áramkörök tervezésével és verifikációjával kapcsolatos lehetőségeket vázolja fel.

A disszertáció első két fejezete a gráfokkal, automatákkal és a modellel kapcsolatos alapvető fogalmakat és eredményeket tárgyalja. Az első fejezet az alapvető halmazelméleti, algebrai, algoritmuselméleti, gráfelméleti és automatelméleti fogalmak tárgyalásával foglalkozik, majd a 2. fejezetben áttekintjük a szoliton automaták analízisével kapcsolatos első eredményeket. A szoliton automata eredeti definíciója egy élsúlyozott gráf modelljén alapult, ezért ebben a részben igazoljuk a mi párosításokon alapuló modellünkkel való ekvivalenciáját.

A 3. fejezettől kezdődően ismertetjük az eredményeinket.

A 3. fejezetben bevezetjük a szétválasztó halmaz fogalmát: akkor mondjuk, hogy a belső csúcsok egy halmaza szétválasztó halmazt alkot a G szoliton gráfban, ha bármely két elemét egy új e éllel összekötve, az e él tiltott lesz $G+e$-ben, azaz $G+e$ egyik állapota sem tartalmazza e-t. A faktor-kritikus gráfok fogalmát is általánosítjuk belső párosításokra: egy összefüggő G gráfot faktor-kritikusnak nevezünk, ha bármely v belső csúcsa esetén G rendelkezik egy olyan párosítással, amely v-t leszámítva minden belső csúcsot lefed.

A fenti új fogalmak felhasználásával két Tutte típusú tételt bizonyítunk a szétválasztó halmazokra. Az első tétel a maximális szétválasztó halmazok jellemzését adja meg, míg a második eredmény azon szétválasztó halmazok egy fontos tulajdonságát írja le, amelyek maximálisak arra vonatkozólag, hogy nem tartalmaznak elérhető csúcsot. (Egy csúcs elérhető, ha létezik hozzá egy külső csúcsból induló alternáló út, amelyik az adott párosítás által tartalmazott élben végződik).

Tétel. *Legyen X a G szoliton gráf belső csúcsainak egy nemüres halmaza, és jelölje $c_{\rm in}(G,X)$ a $G-X$ azon összefüggő komponenseinek a számát, melyek csak belső csúcsokat tartalmaznak. Ekkor a következő két állítás ekvivalens.*

(i) Az X egy maximális szétválasztó halmaz.

(ii) $G-X$ minden olyan összefüggő komponense, amely nem degenerált (egy külső csúcsból áll) faktor-kritikus, valamint

(iia) $|X| = c_{\rm in}(G,X)+1$, vagy

(iib) $|X| = c_{\rm in}(G,X)$ és $G-X$ minden külső csúcsot tartalmazó komponense degenerált.

Továbbá, az (iib) feltétel akkor és csak akkor áll fenn, ha X nem tartalmaz elérhető csúcsot.

Tétel. *Egy külső csúccsal rendelkező G gráf akkor és csakis akkor szoliton gráf, ha a belső csúcsainak bármely X részhalmaza esetén fennáll a $c^{\rm o}_{\rm in}(G,X) \le |X|$ egyenlőtlenség, ahol $c^{\rm o}_{\rm in}(G,X)$ jelöli $G-X$ azon külső csúcs nélküli összefüggő komponenseinek a számát, amelyek páratlan számú belső csúcsból állnak. Egyenlőség akkor és csakis akkor állhat fenn egy nemüres X-re, ha G nem minden összefüggő komponense külső csúcsot tartalmazó faktor-kritikus gráf. Ebben az esetben bármely olyan szétválasztó halmaz biztosítja az egyenlőséget, amely maximális arra vonatkozólag, hogy nem tartalmaz elérhető csúcsot.*

A 4. fejezetben néhány technikai lemma bizonyítása után általánosítjuk az elemi gráfok (olyan gráfok, melyek megengedett élei egy összefüggő feszítő részgráfot alkotnak) kanonikus osztályozását minden olyan gráfra, amely rendelkezik teljes belső párosítással. Ezen osztályozás előáll a maximális szétválasztó halmazoknak az elemi komponensek csúcshalmazaival való metszetei által. Ezen osztályozás alapján az elérhető elemi komponensek (elemi komponensek, amelyek tartalmaznak elérhető csúcsot) egy struktúrába rendezhetőek, amely struktúra tükrözi a komponensek külső csúcsból induló alternáló utak által történő elérhetőségét. Ezen eredmények az alábi tétel összegzi.

Tétel. *Az elérhető elemi komponensek diszjunkt családokba rendezhetőek a következő módon.*

(i) *Bármely $\mathcal{F}$ család legfeljebb egy külső csúccsal rendelkező elemi komponenst (külső elemi komponens) tartalmaz, amely családokra külső családokként hivatkozunk.*

(ii) *Minden $\mathcal{F}$ belső családon (olyan család, amely nem tartalmaz külső komponenst) belül létezik pontosan egy P kanonikus osztály, amelyet bármely olyan külső alternáló út érint, amely az $\mathcal{F}$ valamely eleméhez vezet. Továbbá bármely külső alternáló út ezen úgynevezett principális osztályt érinti először a családon belül.*

(iii) *Egy résznbenrendezés alakítható ki a családok között, amelyben a rendezési relációt az a sorrend definiálja, amint bármely külső alternáló út érinti a családokat. A részbenrendezés maximális elemei a külső családok.*

(iv) *Egy elérhető elemi komponenshez tartozó csúcs akkor és csakis akkor nem elérhető, ha egy principális kanonikus osztályban fekszik.*

A fenti struktúra kialakítása után szétválasztó halmazok és faktor kritikus gráfok segítségével jellemezzük a családokat, majd ezen eredmények felhasználásával megadunk egy algoritmust, amely az élek számában lineáris időben meghatározza bármely G szoliton gráf G^+ járható részgráfját (a külső élek és azon belső élek által feszített részgráfot, amelyeknek legalább az egyik végpontja elérhető), továbbá kialakítja a családokat és a családok közötti részbenrendezést.

Az 5. fejezetben rátérünk az előzőekben meghatározott gráfstruktúra segítségével a szoliton automaták analízisére. Első feladatként definiáljuk az Automata Konstrukciós Problémát (Automaton Construction Problem - ACP) szoliton gráfokra: adott szoliton gráf esetén konstruáljuk meg a gráfhoz rendelt automatát. A feladat megoldása érdekében először jellemezzük a szoliton átmeneteket alternáló vonalak segítségével, amely az ACP egy algoritmikus megoldásához vezet.

Tétel. *Legyen G egy szoliton gráf, és jelölje k a G^+ állapotainak a számát. Ekkor ACP megoldható $\mathcal{O}(k^2 \cdot n \cdot m)$ időben.*

Dassow és Jürgensen egyik cikkükben jellemezték azon determinisztikus szoliton automatákat, melyek egy külső csúccsal rendelkeznek. Ezen eredmény általánosításaként bebizonyítjuk, hogy tetszőleges (nemdeterminisztikus) szoliton automata vagy egy teljes automata (egy inputtal rednelkező automata, amelynek bármely két állapota között létezik átmenet) vagy pedig szemi-teljes (bármely két különböző állapota között létezik átmenet, de önmagához egyik állapotnak sincs átmenete). A fentieken túlmenően jellemezzük a szemi-teljes szoliton automaták gráfstruktúráját is. Ezen eredmény a szoliton automaták dekompozíciójában is központi szerepet játszik, amely dekompozíció az úgynevezett kanonikus szorzaton fogalmán alapul.

Egy kanonikus szorzat egy olyan speciális α_0^ε-szorzat, ahol az automaták két szinten helyezkednek el. Az alsó szinten elhelyezkedő (nem feltétlenül szoliton) automaták az első szinten elhelyezkedő szoliton automatákhoz azok kanonikus osztályain keresztül kapcsolódnak. Ezen kapcsolódás egy úgynevezett kanonikus függőség által definiált, amely az alsó szinten levő automaták halmazából a felső szinten elhelyezkedő szoliton automaták kanonikus osztályainak hatványhalmazába történő leképezés. Egy alsó szinten levő automatában akkor indukálódik egy átmenet, ha az inputként adott csúcspár első komponenséből az automata elérhető a kanonikus függőség által meghatározott valamely kanonikus osztályon keresztül. A fő eredmény formalizálásához bevezetjük az úgynevezett szoliton izomorfizmus fogalmát, amely szerint akkor izomorf két automata egymással, ha a kiterjesztett automatáik (azon automata, amelynél az üres szoliton séta is megengedett) erősen izomorfak.

Tétel. *Jelölje $\mathcal{C}$ azon automaták osztályát, amelyek előállnak az elemi gráfon definiált szoliton automaták (elemi szoliton automaták) egy rendszeréből a teljes automaták egy rendszerébe történő kanonikus szorzat eredményeként. Ekkor a szoliton automaták osztálya szoliton izomorfizmus erejéig egybeesik $\mathcal{C}$-vel.*

Ezen rész zárásaként az Automata Specifikációs Probléma (Automaton Description Problem – ADP) egy analízisét végezzük el. Ebben az esetben csak az az elvárásunk, hogy a szoliton gráf alapján specifikáljuk a megfelelő automatát, azaz adjuk meg egy formális leírását. Az ACP-re adott megoldás természetesen ADP-re is megoldásként szolgál, de az általunk kidolgozott Elemi Struktúrális Kódolás segírtségével belátható, hogy általában a feladat komplexitása jelentősen csökkenthető. A szoliton gráfok Elemi Struktúrális Kódja a fentebb bemutatott elemi dekompozíción túl további kiegészítő információkkal az elemi gráfok szintjén kódolja a szoliton gráfokat, de olyan módon, hogy az ADP szempontjából ez ekvivalens. Ezen struktúrát az ACP-re kapott algoritmikus eredményekkel ötvözve az ADP polinomiális időben való megoldására kapunk egy elegendő feltételt.

Tétel. *Legyen G egy szoliton gráf, melynek minden külső elemi komponense polinomiális számú állapottal rendelkezik, továbbá minden elérhető C belső elemi komponense esetén polinomiális időben meghatározható C állapotainak a száma. Ekkor ADP polinomiális időben megoldható G-re.*

A 6. fejezetben a determinisztikus szoliton automatákkal foglalkozunk. A determinisztikussság fogalma a struktúrális eredményeink alapján természetes módon kiterjeszthető: egy szoliton automata parciálisan determinisztikus, ha az alapobjektumát képező gráf bármely külső elemi komponense egy determinisztikus szoliton gráfot alkot. Először adunk egy párosításokon alapuló jellemzést a determinisztikus szoliton gráfokra: bebizonyítjuk, hogy egy szoliton gráf determinisztikus akkor és csakis akkor, ha a járható részgráfjának minden összefüggő komponense vagy egy olyan gráf, amely nem tartalmaz alternáló kört egyik állapotban sem, vagy egy úgynevezett gesztenyegráf (egy olyan összefüggő gráf, amely egy páros körből, valamint diszjunkt fákból tevődik össze olyan módon, hogy bármely két olyan csúcs között, amelynek fokszáma nagyobb mint 2 a távolság páros). A determinisztikus és parciálisan determinisztikus szoliton gráfok struktúrájának részletes leírása érdekében bevezetünk egy redukciós eljárást a gráfokon, amely a gráfokhoz rendelt automatákon megőrzi az izomorfizmust. Ezen rész központi eredményeként bebizonyítjuk, hogy egy elemi gráf determinisztikus akkor és csakis akkor, ha a bevezetett redukciós módszerrel iteratív módon egy általánosított fára (olyan gráf, amely nem tartalmaz páros hosszú kört) lehet össszehúzni. Ezen tétel eredményeként a determinisztikus és a parciálisan determinisztikus szoliton automaták osztálya leírható gesztenyecsonkok (redukált gesztenyegráfok) és általánosított fák kanonikus s diszjunkt és szorzataként(kvázi-direkt kanonikus szorzatok).

Tétel. *Jelölje $\mathcal{T}$ azon szoliton automaták osztályát, melyek alapobjektuma vagy egy általánosított fa, vagy egy olyan gráf, amely egy élből és az egyik végpontján egy hurokélből áll. Továbbá jelölje $\mathcal{D}$ azon $\mathcal{A}(G)$ szoliton automaták osztályát, melyek vagy $\mathcal{T}$-hez tartoznak vagy G egy gesztenyecsonk. Ekkor a következők állnak fenn.*

(i) A parciálisan determinisztikus szoliton automaták osztálya szoliton izomorfizmus erejéig egybeesik azon automaták osztályával amelyek előállnak $\mathcal{T}$-beli automaták egy rendszeréből a teljes automaták egy rendszerébe történő kanonikus szorzat eredményeként.

(ii) A determinisztikus szoliton automaták osztálya erős izomorfizmus erejéig egybeesik azon automaták osztályával amelyek előállnak $\mathcal{D}$-beli automaták diszjunkt szorzataként.

Az elemi determinisztikus gráfok fenti karakterizációja és az Elemi Struktúrális Kódolás kapcsán kifejlesztett elemi dekompozíciós eljárás ötvözésével egy $\mathcal{O}(n^3)$ futási idejű algoritmust kapunk annak eldöntésére, hogy egy szoliton gráf determinisztikus-e. Ezen algoritmus három eljárásból tevődik össze: az elemi dekompozíció megkonstrulásából, a redukciós módszerből, valamint egy eljárásból, amely a redukát gráfokon a páros kör létezését ellenőrzi.

A disszertációt az eredményeinkkel kapcsolatos nyitott kérdésekkel és jövőbeni kutatási feladatokkal foglalkozó fejezet zárja.

A tézisek erősen támaszkodnak az [11], [15], [17], [18], [19] és [78] cikkeinkre.

Index

Bibliography

[1] A. Adamatzky, *Computing in Nonlinear Media and Automata Collectives*, IOP Publishing, Bristol-Philadelphia, 2001.

[2] A. Adamatzky (Ed.), *Collision-Based Computing*, Springer-Verlag, 2002.

[3] L. M. Adleman, Molecular computation of solutions to combinatorial problems, *Science* **222** (1994), 1021–1024.

[4] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, Massachusettes, 1974.

[5] A. Aviram, M. A. Ratner, Molecular rectifiers, *Chem. Phys. Letters* **29** (1974), 277–283.

[6] H. P. Barendregt, *The Lambda Calculus: Its Syntax and Semantics*, North-Holland, Amsterdam, 1984 (2nd revised edition).

[7] M. Bartha, The Gallai-Edmonds algebra of graphs, *Congressus Numerantium* **123** (1997), 205–219.

[8] M. Bartha, E. Gombás, A structure theorem for maximum internal matchings in graphs, *Information Processing Letters* **40** (1991), 289-294.

[9] M. Bartha, E. Gombás, On graphs with perfect internal matchings, *Acta Cybernetica* **12** (1995), 111-124.

[10] M. Bartha, H. Jürgensen, Characterizing finite undirected multigraphs as indexed algebras, Research Report # 252, Department of Computer Science, The University of Western Ontario, London, Canada, 1989.

[11] M. Bartha, M. Krész, Elementary decomposition of soliton automata, *Acta Cybernetica* **14** (2000), 631-652.

[12] M. Bartha, M. Krész, On the immediate predecessor relationship in soliton graphs, *Congressus Numerantium* **150** (2001), 15–31.

[13] M. Bartha, M. Krész, Characterizing soliton graphs, in *Abstracts of the Fourth Joint Conference on Mathematics and Computer Science* (2001).

[14] M. Bartha, M. Krész, Deterministic soliton automata, *manuscript* (2004).

[15] M. Bartha, M. Krész, Isolating the families of soliton graphs, *Pure Mathematics and Applications* **13** (2002), 49–62.

[16] M. Bartha, M. Krész, Soliton graphs and graph-expressions, in *Abstracts of the 3rd CSCS Conference* (2002).

[17] M. Bartha, M. Krész, Structuring the elementary components of graphs having a perfect internal matching, *Theoretical Computer Science* **299** (2003), 179–210.

[18] M. Bartha, M. Krész, Deterministic soliton automata defined by elementary graphs, in *Proceedings of the Kalmár Workshop on Logic and Computer Science* (2003) pp. 69–79.

[19] M. Bartha, M. Krész, Tutte type theorems in graphs having a perfect internal matching, *Information Processing Letters* **91** (2004), 277–284.

[20] M. Bartha, M. Krész, Redex elimination in graphs, in *Abstracts of the Fifth Joint Conference on Mathematics and Computer Science* (2004).

[21] M. Bartha, M. Krész, The Berge formula, and a calculus of barriers for maximum internal matchings in graphs, in *Abstracts of the Graph Theory 2004 Conference* (2004).

[22] M. Bartha, M. Krész, Algorithms for soliton automata, *manuscript* (2004).

[23] M. Bauderon, B. Courcelle, Graph expressions and graph rewritings, *Math. Systems Theory*, **20** (1987), 83–127.

[24] C. Berge, Sur le couplage maximum d'un graphe, *C. R. Acad, Sci. Paris Sér. I Math.* **247** (1958), 258–259.

[25] F. L. Carter, Comformational switching at the molecular level, in *Molecular Electronic Devices* (F. L. Carter Ed.), Marcel Dekker, Inc., New York, 1982, pp. 51–72.

[26] F. L. Carter (Ed.), *Molecular Electronic Devices*, Marcel Dekker, Inc., New York, 1982.

[27] F. L. Carter, The molecular device computer: Point of departure for large scale cellular automata, *Physica D* **10** (1984), 175–194.

[28] F. L. Carter (Ed.), *Molecular Electronic Devices II*, Marcel Dekker, Inc., New York, 1987.

[29] J. Cheriyan, A. Sebő, Z. Szigeti, Improving on the 1.5-approximation of a smallest 2-edge connected spanning subgraph, *SIAM J. Discrete Math.* **14** (2001), 170–180.

[30] J. Cheriyan, R. Thurimella, Approximation algorithms for minimum-cost k-connected subgraphs via matching, *SIAM J. Comput.* **30** (2000), 528–560.

[31] M. Conrad, Molecular computing: The lock and key pardigm, *IEEE Computer* **25** (1994), 11–20.

[32] W. H. Cunningham, J. F. Geelen, The optimal path-matching problem, *Combinatorica* **17** (1997), 315–336.

[33] A. Czumaj, A. Lingas, On approximability of the k-connected spanning subgraph problem, in *Proc. 10th Annual ACM-SIAM Symposium on Discret. Alg. (SODA)*, (1999), pp. 281–290.

[34] J. Dassow, H. Jürgensen, Soliton automata, *J. Comput. System Sci.* **40** (1990), 154-181.

[35] J. Dassow, H. Jürgensen, Soliton automata with a single exterior node, *Theoretical Computer Science* **84** (1991), 281–292.

[36] J. Dassow, H. Jürgensen, Soliton automata with at most one cycle, *Journal of Computer and System Sciences* **46** (1993), 155–197.

[37] J. Dassow, H. Jürgensen, The transition monoids of soliton trees, Research Report # 331, Department of Computer Science, The University of Western Ontario, London, Canada, 1992.

[38] A. L. Dulmage, N. S. Mendelsohn, Coverings of bipartite graphs, *Canad. J. Math.* **10** (1958), 517–534.

[39] A. L. Dulmage, N. S. Mendelsohn, A structure theory of bipartite graphs of finite exterior dimension, *Trans. Roy. Soc. Canada Ser. III* **53** (1959), 1–13.

[40] J. Edmonds, Paths, trees and flowers, *Canad. J. Math.* **17** (1965), 449–467.

[41] J. Edmonds, Matroid partition, in *Math. Decision Sciences, Part I., Lectures in Appl Math. 11* (1968), 315–336.

[42] Z. Ésik, Homomorphically complete classes of automata with respect to the α_2-product, *Acta Sci. Math.* **48** (1985), 135–141.

[43] Z. Ésik, J. Virágh, On products of automata with identity, *Acta Cybernetica* **7** (1986), 299–311.

[44] R. P. Feynman, There's plenty of room at the bottom, in *Miniaturization* (H. D. Gilbert Ed.), Reinhold, New York, 1961, pp. 282.

[45] A. Frank and L. Szegő, A note on the path-matching formula, *Journal of Graph Theory* **41** (2002) 110–119.

[46] C. G. Fernandes, A better approximation for the minimum k-edge connected spanning subgraph problem, *J. Algorithms* **28** (1998), 105–124.

[47] G. Frobenius, Über zerlegbare Determinanten, *Sitzungsber. König. Preuss. Akad. Wiss.* **XVIII** (1917), 274–277.

[48] H. N. Gabow, An efficient implementation of Edmonds' algorithm for maximum matching on graphs, *J. Assoc. Comput. Mach.* **23** (1976), 221–234.

[49] H. N. Gabow, R. E. Tarjan, A linear-time algorithm for a special case of disjoint set union, *Journal of Computer and System Sciences* **30** (1985), 209–221.

[50] H. N. Gabow, R. E. Tarjan, Faster scaling algorithms for general graph-matching problems, *J. Assoc. Comput. Mach.* **38** (1991), 815–853.

[51] H. N. Gabow, H. Kaplan, R. E. Tarjan, Unique maximum matching algorithms, *Journal of Algorithms* **40** (2001), 159–183.

[52] T. Gallai, Neuer Beweis eines Tutte'schen Satzes, *Magyar Tud. Akad. Mat. Kutató Int. Közl.* **8** (1963), 135–139.

[53] T. Gallai, Kritische Graphen II, *Magyar Tud. Akad. Mat. Kutató Int. Közl.* **8** (1963), 373–395.

[54] T. Gallai, Maximale Systeme unabhängiger Kanten, *Magyar Tud. Akad. Mat. Kutató Int. Közl.* **9** (1964), 401–413.

[55] F. Gécseg, Composition of automata, in *Automata, Languages and Programming*, 2nd Colloquium, Saarbrücken, 1974, Lecture Notes in Computer Science, Vol.14, Springer-Verlag, Berlin-Heidelberg-New York-Tokyo, 1974, pp. 351–363.

[56] F. Gécseg, *Products of Automata*, Akademie-Verlag, Berlin, 1986.

[57] F. Gécseg, B. Imreh, On completeness of nondeterministic automata, *Acta Math. Hungar.* **68** (1-2) (1995), 151-159.

[58] F. Gécseg, H. Jürgensen, Automata represented by products of soliton automata, *Theoretical Computer Science* **74** (1990), 163–181.

[59] M. P. Groves, Towards verification of soliton circuits, in *Molecular Electronic Devices* (F. L. Carter, R. E. Siatkowski, H. Wohltjen Ed.), North-Holland, Amsterdam, 1988, pp. 287–302.

[60] M. P. Groves, C. F. Carvalho, C. D. Marlin, and R. H. Prager, Using Soliton Circuits to Build Molecular Memories, *Australian Computer Science Communications* **15** (1993) pp. 37–45.

[61] M. P. Groves, C. D. Marlin, Using Soliton Circuits to Build Molecular Computers, *Australian Computer Science Communications* **17** (1995) pp. 188–193.

[62] M. P. Groves, C. F. Carvalho, and R. H. Prager, Switching the polyacetylene soliton, *Materials Science and Engineering* **C3** (1995) pp. 181–185.

[63] M. P. Groves, Soliton Circuit Design Using Molecular Gate Arrays, in *Proceedings of the 20th Australasian Computer Science Conference* (1997) pp. 245–252.

[64] M. Grötschel, L. Lovász, A. Schrijver, The ellipsoid method and its consequences in combinatorial optimization , *Combinatorica* **1** (1981), 169–197.

[65] P. Hall, On representatives of subsets, *J. London Math. Soc.* **10** (1935), 26–30.

[66] G. Hetyei, Rectangular configurations which can be covered by 2×1 rectangles, *Pécsi Tan. Föisk Közl.* **8** (1964), 351–367. (Hungarian)

[67] J. Heuvel, M. Johnson, The external network problem with vertex-connectivity requirements, in *Abstracts of the Graph Theory 2004 Conference* (2004).

[68] B. Imreh, M. Ito, On α_i-product of nondeterministic automata, *Algebra Colloquium* **4** (1997), 195–202.

[69] A. Itai, M. Rodeh, S. Tanimoto, Some matching problems for bipartite graphs, *Journal of the ACM* **25** (1978), 517–525.

[70] P. W. Kasteleyn, Dimer statistics and phase transitions, *J. Math. Phys.* **4** (1963), 287–293.

[71] P. W. Kasteleyn, Graph theory and crystal physics, in *Graph Theory and Theoretical Physics* (F. Harary Ed.), Academic Press, New York, 1967, pp. 43–110.

[72] D. E. Knuth, *The Art of Computer Programming. Vol.1: Fundamental Algorithms*, 2nd e., Addison-Wesley, Reading, MA, 1973.

[73] D. König, Line systems and determinants, *Math. Termész. Ért.* **33** (1915), 221–229. (Hungarian)

[74] A. Kotzig, On the theory of finite graphs with a linear factor I, *Mat-Fyz. Casopis Slovensk. Akad. Vied* **9** (1959), 73–91. (Slovak)

[75] A. Kotzig, On the theory of finite graphs with a linear factor II, *Mat-Fyz. Casopis Slovensk. Akad. Vied* **9** (1959), 136–159. (Slovak)

[76] A. Kotzig, On the theory of finite graphs with a linear factor III, *Mat-Fyz. Casopis Slovensk. Akad. Vied* **10** (1960), 205–215. (Slovak)

[77] M. Krész, *Designing efficient internal matching algorithms*, Master Thesis, Department of Informatics, József Attila University, Szeged, Hungary, 1994. (Hungarian)

[78] M. Krész, Alternating cycles in soliton graphs, *WSEAS Transactions on Mathematics* **1** (2002), 165–170.

[79] M. Krész, S-matchings, *manuscript* (2004).

[80] M. Krész, Isomorphically complete systems of soliton automata, *manuscript* (2004).

[81] K. Kuratowski, Sur la probléme des courbes gauches en topologie, *Fund. Math.* **16** (1930), 271–283.

[82] C. H. Little, Kasteleyn's theorem and arbitrary graphs, *Canad J. Math.* **25** (1973), 758–764.

[83] L. Lovász, On the structure of factorizable graphs, *Acta Math. Acad. Sci. Hungar.* **23** (1972), 179–195.

[84] L. Lovász, On the structure of factorizable graphs II, *Acta Math. Acad. Sci. Hungar.* **23** (1972), 465–478.

[85] L. Lovász, M. D. Plummer, *Matching Theory*, North Holland, Amsterdam, 1986.

[86] M. Micali, V. V. Vazirani, An $\mathcal{O}(\sqrt{|V|}|E|)$ algorithm for finding maximum mathing in general graphs, in *Proc. 21st Anual IEEE Symposium in Foundation of Computer Science*, 1980, pp. 17–27.

[87] T. Sienko, A. Adamatzky, N. Rambidi, and M. Conrad (Ed.), *Molecular Computing*, The MIT Press, London, 2003.

[88] D. M. Taylor, Molecular nanostructures and their electrical probing, *Thin Solid Films* **331** (1998), 1–7.

[89] C. H. Papadimitriou, *Computational Complexity*, Addison-Wesley Publishing Company, Inc., 1994.

[90] B. Spille and L. Szegő, A Gallai-Edmonds-type structure theorem for path-matchings, *Journal of Graph Theory* **46**(2004), 93–102.

[91] B. Spille and R. Weismantel, A combinatorial algorithm for the path matching problem, *manuscript* (2003).

[92] R. E. Tarjan, Depth-first search and linear graph algorithms, *SIAM J. Comput.* **1** (1972), 146–160.

[93] R. E. Tarjan, Finding dominators in directed graphs, *SIAM J. Comput.* **3** (1974), 62–89.

[94] R. E. Tarjan, *Data structures and network algorithms*, Society of Industrial and Applied Mathematics, Philadelphia, Pennsylvania, 1983.

[95] A. Turing, On computable numbers, with an application to the Entscheidungs problem *Proc. London Math. Soc.* **2-42** (1936), 230–265. Correction, ibid., **2-43**, 544–546.

[96] N. Trinajstic, *Chemical graph theory*, CRC Press, Boca Raton, Florida, 1983.

[97] W. T. Tutte, The factorization of linear graphs, *J. London Math. Soc.* **22** (1947), 107–111.

[98] W. T. Tutte, The factors of graphs, *Canad. J. Math.* **4** (1952), 314–328.

[99] L. G. Valiant, The complexity of computing the permanent, *Theoretical Computer Science* **8** (1979), 410–421.

[100] H. Wielandt, *Finite permutation groups*, Academic Press, New York, 1964.

[101] Y. M. Yin, X. Q. Lin, Progress on molecular computer, *Progress in Chemistry* **13** (2001), 337–342.