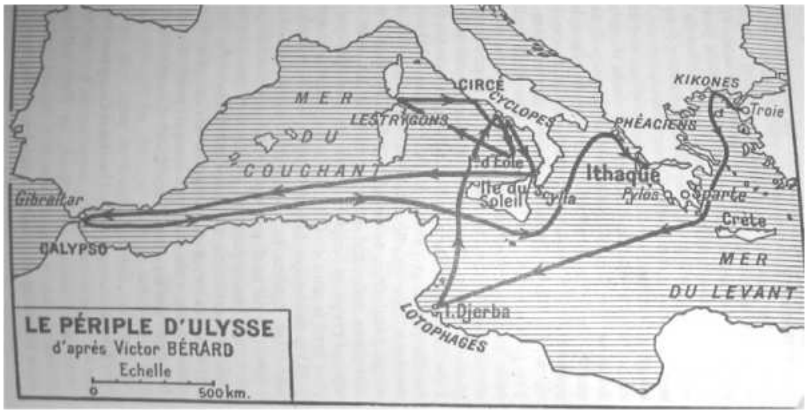
The Odyssey of MSC-Graphs

Blaise Genest



14th September 2004

Contents

List of Figures

Introduction

The Goals

Communicating protocols are trading rules between different processes. With the development of communicating nets, like Internet and phone nets, the protocols used are increasingly complex. For instance, money transactions on Internet are set by strict protocols. In this thesis, we will consider asynchronous communications by means of message passing.

Understanding. The main problem with asynchronous protocols is that they are more complex to understand than synchronous ones. Hence, one of the first task is to understand the kind of protocols that can be described with one asynchronous structures or another one. This work has begun with [HMNK$^+$04] for a restricted class of protocols, then slightly extended by [Kus03, Mor02]. Lastly, [BL04] have obtained partial results without restricting the protocols.

Building. Communicating protocols work on parallel processes, that is each process makes choices depending only upon its own knowledge (states), and not upon the state of the whole system. We will say that such processes have a *local control*. A shared belief is that the human brain works intrinsically sequentially, and that thinking in a parallel way is much harder. Hence, another goal is to help designing communicating protocols, for instance in distributing automatically algorithms that are too sequential into algorithms with a local control, in a way described in [AEY00, AEY01]. However, they assumed that no additional (control) data is allowed. Another approach was used by [HJ00], who exhibited a class of protocols that are always implementable, and by [CDHL00] who supposed that more freedom is left to engineers.

Verifying. At last, the complexity of the protocols, together with the complexity of parallel algorithms has led to huge mistakes in the past. An early verification of communicating protocols would save much money. This idea was first expressed by [AY99, MP99], who modeled the property by a set of executions to forbid, then by [Pel00, MM01] who considered a logical property upon partial orders.

The Tools

The tools that we use here are the following ones:

Automata is a well studied structure by computer scientists. We will heavily use algorithms about automata, and more generally about regular languages. Hence, we will try to express the systems and properties by means of regular languages. For more information about automata theory, see [HU79].

Decidability Knowing algorithms that compute what we want is necessary, but not sufficient. The first problem is that some algorithms might not always terminate, that is never give an answer.

Complexity Moreover, even if an algorithm always gives a right answer, the time needed might be practically out of reach. When a question is undecidable, it usually suffices to reduce that question to a subclass. This subclass should allow expressing interesting models and should have a much better complexity. Hence, an algorithm is usually not sufficient to solve a problem. Several algorithms are better: A fast algorithm for a really restricted class, and slower ones for more general classes of languages. For more information on complexity, see [Pap94, GJ78].

Mazurkiewicz Traces Communicating protocols use concurrent (parallel) processes. Actually, Mazurkiewicz Traces model precisely concurrent events. It would be fruitful to discover the classes of protocols for which Mazurkiewicz Trace theory and its numerous results can be applied to communicating protocols. For more information, see [DR95].

Verification At last, to verify that a protocol satisfies some specification, we will use the theory of Model-Checking. That is, we will consider that the system is given by an abstract model, and that the property is given by a logical formula or by the set of executions of another model. The aim is to know whether every execution of the model is part of the property (positive model-checking), or if none of the executions of the model intersects those from the property (negative model-checking). The models considered can be too large to be directly verified. So they should be abstracted in smaller models, getting rid of the useless details for the property to verify. For more information about verification, see [CGP00, BBF$^+$01].

To describe communicating scenarios, a widely used model, *time-space diagrams* [LL90, Lyn96], was reintroduced as *Message Sequence Chart (MSC)* by UML (Unified Modeling Language) [BJR97]. In order to describe a whole communicating protocol, nets of *communicating automata (CFM)* [BZ83] are used. Another possibility is to use a kind of MSC automata called MSC-graphs [itu96].

This PhD thesis is separated into two parts, the part Automata, Communication and Diagrams, and the part Verification. The first part focuses on communicating automata (understanding their expressivity etc.), and tries to enlighten them with others structures, like MSC-graphs. The second part focuses on MSC-graphs. Because of their visual aspect, they are easier to understand. Moreover, the global control of MSC-graphs allows defining decidable restrictions.

Outline of the Part Automata, Communication and Diagrams

The aim of this part is to show the relations between several structures communicating by messages. We begin by defining the tools we will need in the first three sections of the first chapter: automata and regular languages, decidability and complexity, then Mazurkiewicz traces. We give the theorems binding these structures together: theorems of Kleene, Büchi and Ochmanski. Then we introduce in the last sections of the first chapter communicating structures: logics, communicating automata (that have a local control) [BZ83] and MSC-graphs (that have a global control) [MR97, GMP04]. In particular, we express that modeling communication with messages by Mazurkiewicz traces is difficult in the general setting.

In the second chapter, we explain the relations between communicating automata and other communicating structures. In order to do that, we need the structure of CMSC-graphs, described in [GMP01, GMP04]. We end the chapter with the proof of a theorem explaining the expressivity of communicating automata by means of second order logics and of MSC-graphs, in a slightly restricted case (existentially-bounded communication channels). Actually, this restriction allows to model the communication with messages by traces, which allows to apply theorems for traces, as was done by [Kus03] for universally-bounded channels. In this respect, we extend in [GKM04] results from [HMNK+04, Mor02, Kus03].

Outline of the Part Verification

It is easier to write a MSC-graph modeling a language than a communicating automaton. Hence, the first use of MSC-graphs is to help implementing a protocol. In the third chapter, we raise the question when and how it is possible to transform a given MSC-graph into a deadlock-free communicating automaton. It corresponds to distributing the global control of an MSC-graph into the local control of a CFM. We give an effective solution to that problem in [GMSZ02, GM03, Gen04], where we ask whether a protocol can be expressed using the class of local-choice CMSC-graphs, that are always implementable without deadlock. Unfortunately, this solution is incomplet. Others approaches, also incomplete, can be found in [AEY00, AEY01, BM03].

Hence, MSC-graphs appear early in the modeling of communicating protocols. Verifying protocols at this early stage would then allow to save the implementation time for buggy protocols [AHP96]. Moreover, the details usually do not appear as early in the modeling process, which prevents huge systems. We give the exact complexity of the verification of MSC-graphs against numerous formalisms in the fourth chapter. This work is based on previous work of [MP99, AY99, MM01, Pel00], together with our work [GMSZ02, GKM04, GMMP04].

Anyway, the size of the systems can be so big that algorithms cannot give an answer in a tractable time. Hence, in the fifth chapter, we are interested in reducing the size of MSC-graphs, in the same way as [AY98, AKY99]. We first propose to abstract some parts that are useless for the property to verify. We also propose to use the modular aspect of protocols to save some computing time. Actually, macros express a redundancy inside the protocol (a modular description can be seen as a compression). We show that it is sometimes possible and sometimes hard, to use the redundancy to prevent the computation of the same information several times, and so to speed up algorithms. These aspects were not much considered for MSC-graphs, except in [GM02, GHM03]. However, a connected topic was considered by [ACE$^+$03], who tried to compress communicating scenarios in a modular way.

Part I

Automata, Communication and Diagrams.

Chapter 1

Several Structures on Words

οὕτως οὐχ ἅμα πάντα θεοὶ χαρίεντα διδοῦσιν
ἀνδράσιν, οὔτε φυὴν οὔτ᾽ ἂρ φρένας οὔτ᾽ ἀγορητύν[1]
Homer in The Odyssey, Book VIII, verse 167-168

1.1 Regular Languages

1.1.1 A Machine Model

We define automata on words.

1.1.2 Different Representations

We define rational languages, MSO formula, FO formula and LTL formula over words.

Theorem 1 (Kleene-Büchi[Kle56, Büc60]) *MSO formulae, regular languages and rational expressions have the same expressive power over the free monoid.*

[1] the gods don't hand out all their gifts at once,
not build and brains and flowing speech to all.

1.2 Turing Machines

1.2.1 Definition

1.2.2 Decidability and Complexity

1.3 Mazurkiewicz Traces

We define Mazurkiewicz traces, together with regular and rational traces languages.

1.3.1 Ochmański Theorem

We define loop I-connected automata.

Theorem 2 *[Och85] Let L be a rational trace language over (Σ, I). Then L is regular if and only if L is accepted by a loop I-connected automaton.*

Proposition 1 *If $\mathcal{A}$ is a loop I-connected automaton, then there exists an I-diamond automaton $\mathcal{B}$ such that $[L(\mathcal{A})]_I = L(\mathcal{B})$, and $|\mathcal{B}| = O((2^{|\Sigma|}|\mathcal{A}|^2)^{|\mathcal{A}||\Sigma|})$ [MP99].*

If $\mathcal{A}$ is an automaton with $L(\mathcal{A})$ closed under I-commutation, then there exists a loop I-connected automaton $\mathcal{B}$ such that $[L(\mathcal{B})]_I = L(\mathcal{A})$, and $|\mathcal{B}| = O(|\Sigma|! \times |\mathcal{A}|)$ [Och85] (see [Gen04] for complexity).

We define here MSO, FO formulae on traces, together with local [DG04, GK03] and global [TW02, DG02, Wal98] temporal logics on traces.

1.3.2 Traces and Communication

We explain that communication (through unbounded channels) is a priori hard to model with traces.

1.4 Communicating Machines

We define here communicating automata (CFM), with and without deadlocks.

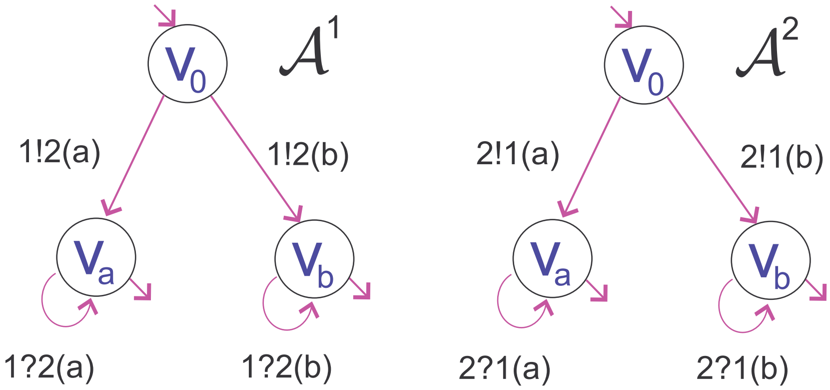


Figure 1.1: A CFM $(\mathcal{A}^1, \mathcal{A}^2)$ with deadlock.

1.4.1 Message Sequence Chart (MSC)

We define here MSCs [itu96], and the universal and existential bound on communication [LM04].

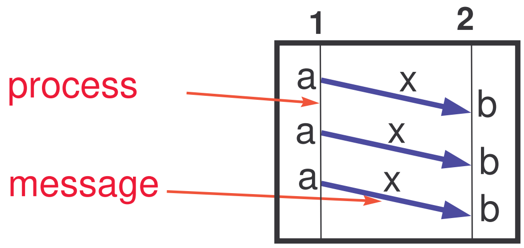


Figure 1.2: A Message Sequence Chart (MSC)

1.4.2 Other Representations for Communication

We define MSO over MSCs, and regular sets of MSCs.

Theorem 3 *[HMNK$^+$04] Let $\mathcal{M}$ be a language of universally bounded MSCs. Then the following propositions are equivalent:*

- $\mathcal{M}$ *is a regular set of MSCs.*
- $\mathcal{M}$ *is the language of some CFM.*
- $\mathcal{M}$ *is the language of an MSO formula.*

1.5 MSC-graphs

We define here MSC-graphs [itu96], which correspond to rational MSC languages.

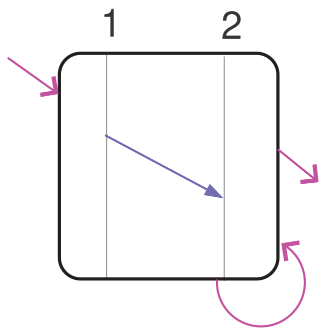


Figure 1.3: An MSC-graph

We define atoms and finite generation.

Proposition 2 *Every finitely generated language of MSCs is existentially bounded.*

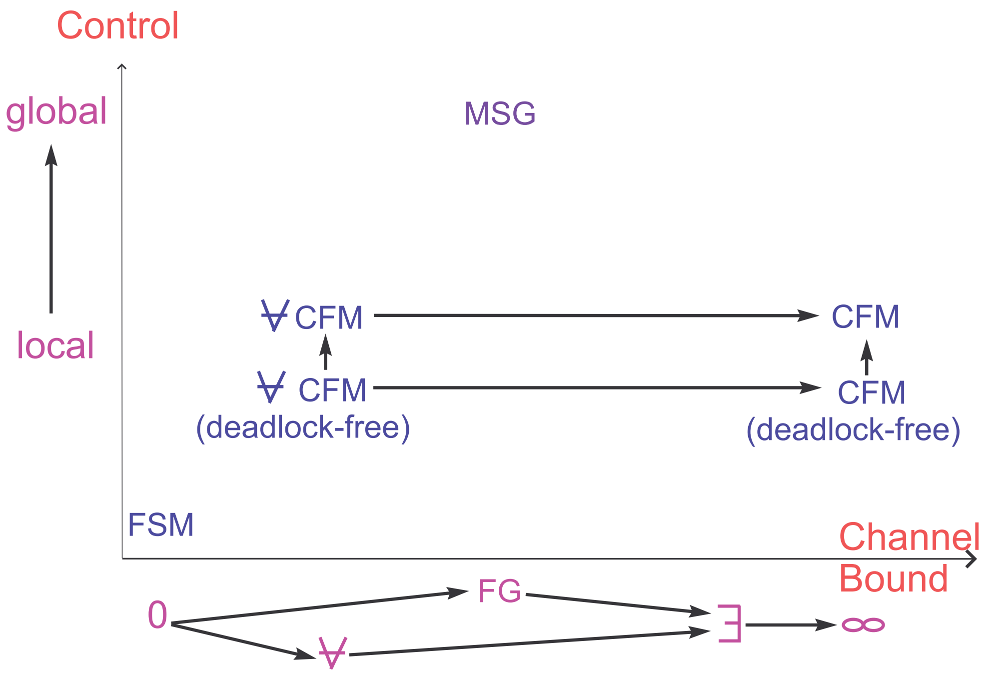


Figure 1.4: Here is the first diagram comparing MSC-graphs (called here MSG) and CFMs. We denote word automata by FSM. The two directions considered are channel bounds, that can be either no communication (0), a universal bound (∀), an existential bound (∃), or no bound (∞). We also denote that a structure is finitely generated by FG. We compare the structures also with respect to their control, which can be local (with or without deadlock) or global. Generally, structures above other structures are more expressive. For the horizontal direction, finitely generated structures and universally bounded structures cannot be compared. To help the reader, we add simple arrows to express a strict inclusion. We denote by an equality the fact that two structures are expressively equivalent.

Chapter 2

One Theorem to Bind Them All

Ash nazg durbatulûk,
Ash nazg gimbatul,
Ash nazg thrakatulûk
agh burzum-ishi krimpatul.
J.R.R. Tolkien

The whole chapter corresponds to the paper "A Kleene Theorem for a Class of Communicating Automata (CFM) with Effective Algorithms" with Dietrich Kuske and Anca Muscholl [GKM04].

2.1 Comparing MSC-graphs and Communicating Automata

2.1.1 CMSC-Graphs

To get rid of finite generation, we define Compositional MSCs and Compositional MSC-graphs (CMSC-graphs) [GMP01, GMP04].

2.1.2 A General Model

Proposition 3 *[GMP01] CMSC-graphs are more expressive than CFM and MSC-graphs.*

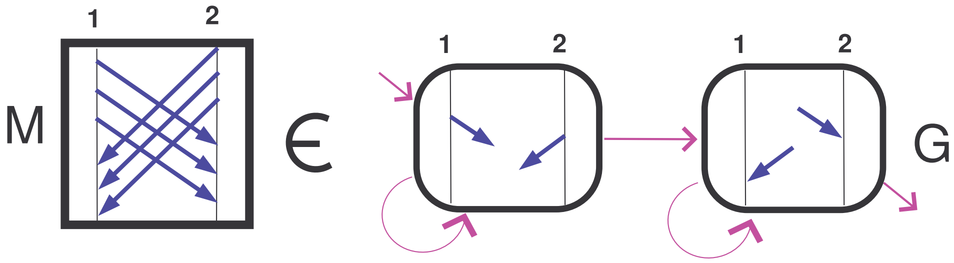


Figure 2.1: A CMSC-graph G.

2.2 Existential Bound

Definition of $\exists$-b-CFM [GKM04] and of safe CMSC-graphs (called realizable in [GMP01] and simply CMSC-graphs in [MM01]). Definition of $Lin^b(\mathcal{M})$, the set of b-bounded linearizations of a language of MSCs $\mathcal{M}$.

2.3 Process Cooperation

Définition 1 *[GKM04] (see also [GMSZ02] for globally cooperative MSC-graphs).*

The communication graph of a CMSC M is a directed graph whose vertices are the processes involved in M, and there is an edge between vertices p, q iff M contains both a send $p!q$ from p to q and a receive $q?p$ on q from p (these send and receive may not define the same message).

*A CMSG is globally-cooperative (*gc-CMSG *for short) if it is safe and every loop is labeled by a CMSC whose communication graph is weakly connected.*

2.3.1 Kuske's Alphabet

We show how to use Kuske's trace alphabet [Kus03] to model communication in existentially B-bounded structures [GKM04]

Proposition 4 *[GKM04]*

If a CFM $\mathcal{A}$ is existentially b-bounded, then an equivalent globally cooperative CMSC-graph of size $O((b\wp^2)!(b\mathcal{A})^{\wp})$ can be built.

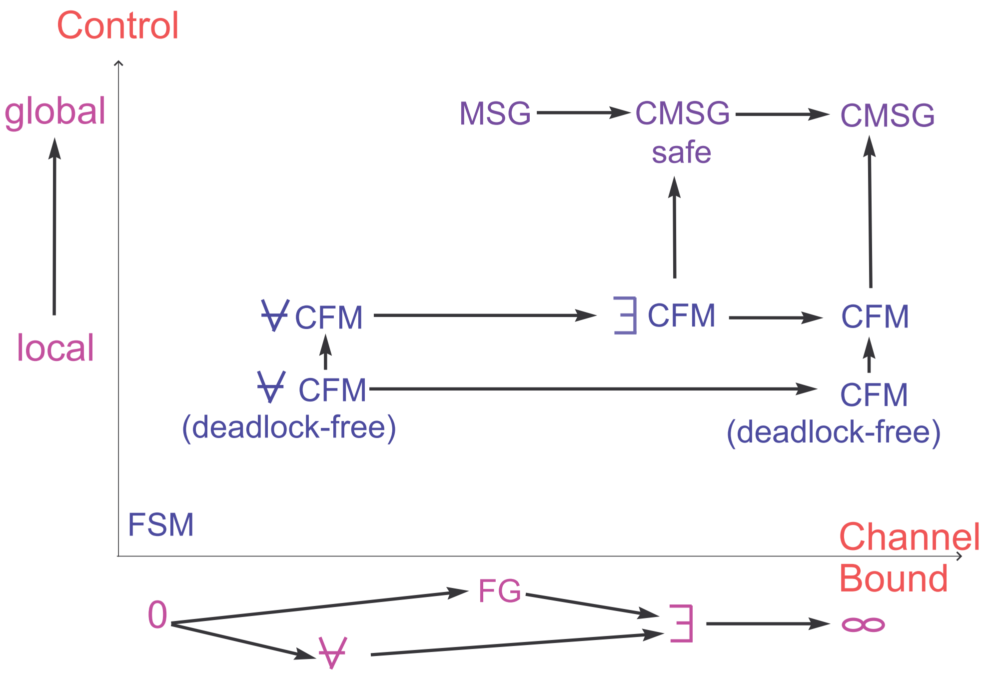


Figure 2.2: Diagram with CMSC-graphs and existentially bounded structures.

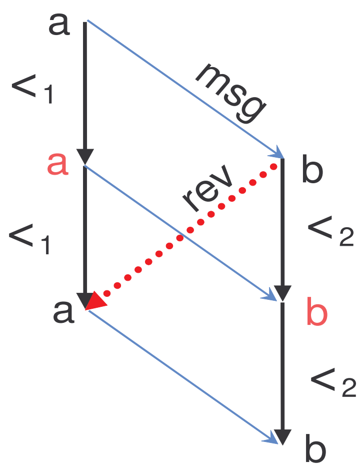


Figure 2.3: Example of a pomset associated with Kuske's alphabet.

Proposition 5 *[GKM04] Let G be a globally-cooperative CMSC-graph. Then $Lin^b(G)$ is regular, accepted by an automaton of size $|G|^{O(b^2\wp^4|G|)}$.*

2.3.2 Regular CMSC-graphs

We can define regular CMSC-graphs as globally cooperative CMSC-graphs that are universally bounded, or with the definition from [MP99, AY99]. Using [HMNK$^+$04], we can state:

Proposition 6 *Let $\mathcal{M}$ be a universally bounded language of MSC. Then the following are equivalent:*

- *$\mathcal{M}$ is regular.*
- *$\mathcal{M}$ is the language of a CFM.*
- *$\mathcal{M}$ is the language of an MSO formula.*
- *$\mathcal{M}$ is the language of a globally cooperative CMSC-graph.*

2.4 Extension of the Kleene-Büchi Theorem

Theorem 4 *[GKM04] Let $\mathcal{M}$ be an existentially bounded language of MSCs. Then the following assertions are equivalent:*

1. *$Lin^b(\mathcal{M})$ is a regular set of representatives for some b.*
2. *$\mathcal{M}$ is the language of a CFM.*
3. *$\mathcal{M}$ is the language of a globally cooperative CMSC-graph.*
4. *$\mathcal{M}$ is the language of an MSO formula.*

The following sections are part of the proof of this theorem.

2.4.1 From MSO to a rRgular Set of Representatives

2.4.2 Zielonka's Theorem

Theorem 5 *[Zie87] A trace language is accepted by an asynchronous mapping if and only if it is regular.*

2.4.3 A CFM Close to Asynchronous Mappings

2.4.4 τ-Cycles

2.4.5 A CFM Accepting $\mathbb{MSC}^b$

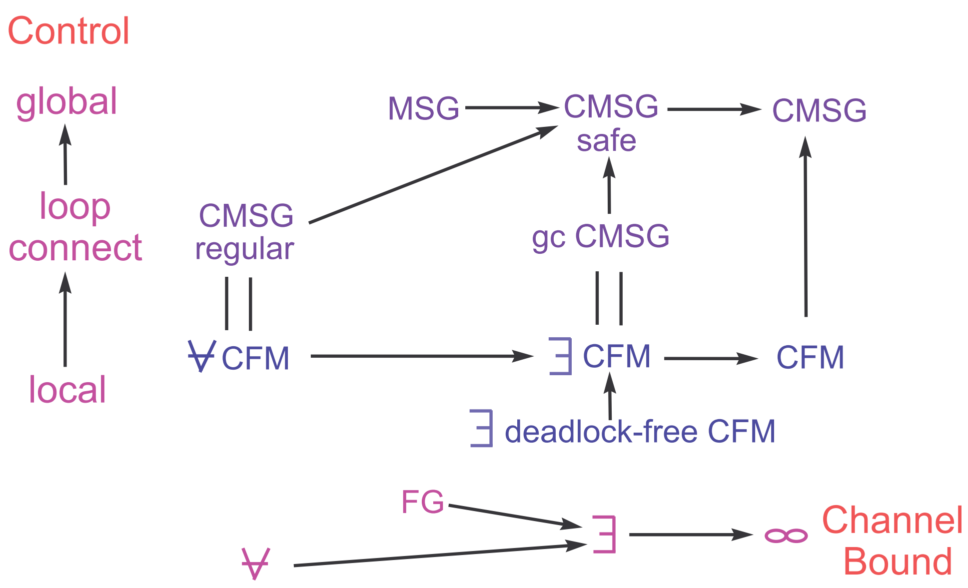


Figure 2.4: Diagram with regular and globally cooperative (gc) CMSC-graphs

Part II

Verification of MSC-Graphs

Chapter 3

Implementation

Great is the strength of feeble arms combined
Homer

The results of this chapter can be found in the paper " The Structure of Local-Choice in High-Level Message Sequence Charts (HMSC)" with Anca Muscholl [GM03]. The implementation of local-choice MSC-graphs can be found in section 5.2 of the journal version of "Infinite-State High-Level MSCs: Model-Checking and Realizability" with Anca Muscholl, Helmut Seidl and Marc Zeitoun [GMSZ02]. The test to know whether a CMSC-graph is equivalent to some local-choice CMSC-graph can be found in chapter 4 of "Compositionality Everywhere: The end of MSC-Graphs?" [Gen04].

3.1 Local-Choice and Implementation

3.1.1 Definition

We define local-choice CMSC-graphs [Gen04] (see also [GM03, GMSZ02] for local-choice MSC-graphs), which is a slight modification of the original definition by [HJ00].

Définition 2 *An MSG (resp CMSG)* $G = \langle V, R, v^0, V_f, \lambda\rangle$ *is called* local-choice *(*lc-(C)MSG *for short) if each (C)MSC labeling a node* v *of* G *is a* **Triangle***, that is it has a single minimal event* $\min(v)$ *for the partial order* $\leq$*. Moreover,* $\min(w)$ *must belong to the process set* $P(v)$ *of node* v*, whenever* $v \to w$*. A lc-CMSG is also required to be safe.*

Proposition 7 *[GMSZ02] Let G be a local-choice CMSC-graph. Then G is implementable without deadlock. The size of the equivalent CFM is linear in the size of the CMSC-graph.*

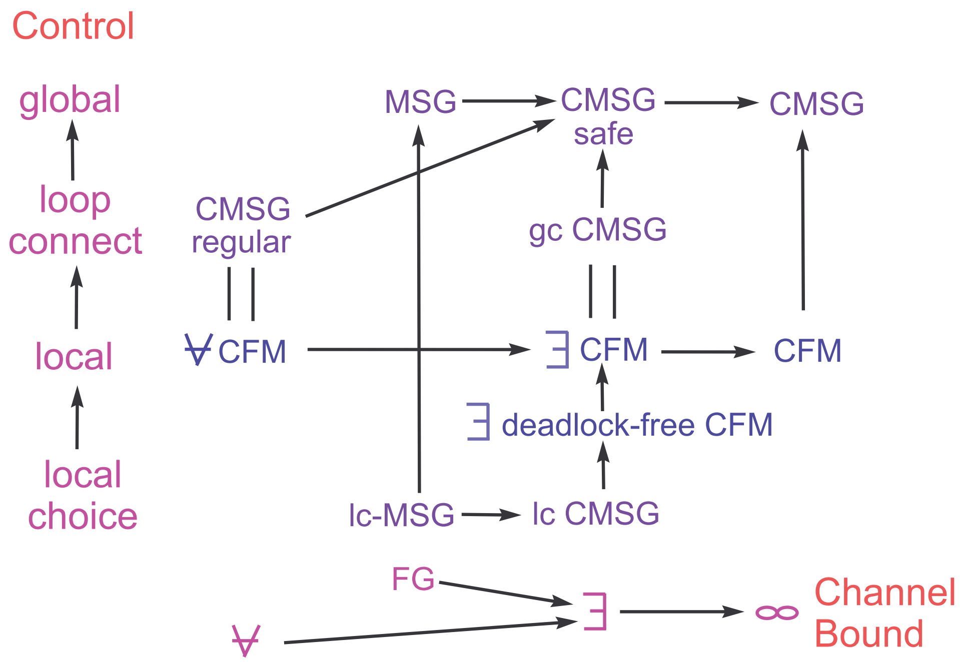


Figure 3.1: Diagram with local-choice MSC-graphs.

3.1.2 Some Concrete Protocols

We describe the protocol *WRS* (see [SC02, GMMP04] and the protocol *USB* (Universal Serial Bus), see [usb96, Gen04].

3.2 Expressivity

3.2.1 Characterization of Local-Choice MSC-graphs

We define cut events, triangles, MSC triangles, pair of parallel MSC in a CMSC-graph, and triangles without cut events (see [GM03]).

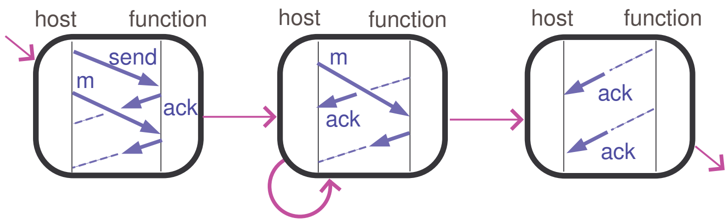


Figure 3.2: Local-choice CMSC-Graph specifying bulk transactions of usb 1.1.

Theorem 6 *[GM03] Let G be a safe CMSC-graph. Then G is equivalent to a local-choice MSC-graph if and only if there exists some integer b such that:*

1. *G is globally cooperative.*
2. *Each $M \in \mathcal{L}(G)$ is a triangle.*
3. *Each G-MSC triangle without cut events M satisfies $|M| \leq b$.*
4. *Each pair of parallel MSCs M, N in G satisfies either $|M| \leq b$, or $|N| \leq b$.*

3.2.2 Two Algorithms for Implementation

Proposition 8 *[GM03] We can effectively test the properties of theorem 6 in PSPACE (co-NP-hard). Moreover, if these tests are positive, then one can build a local-choice MSC-graph equivalent to G, at most doubly exponential in the size of G.*

Theorem 7 *[Gen04] Let G be a safe CMSC-graph. Then it is decidable in co-NP whether G is equivalent to a local-choice CMSC-graph. If the answer is positive, then an equivalent local-choice CMSC-graph can be built of size at most exponential in $|G|$.*

Chapter 4

Model-Checking

Felix qui potuit rerum cognoscere causas [1]
Virgile in The Georgics, Book II, verse 489

This chapter uses the results of [GKM04] for the model-checking of existentially bounded structures. Results concerning the model checking of local-choice (C)MSC-graphs can be found in chapter 4.4 of the journal version of [GMSZ02]. The section concerning template MSCs corresponds to the paper "Specifying and Verifying Partial Order Properties using Template MSCs" with Marius Minea, Anca Muscholl and Doron Peled [GMMP04].

4.1 Logics

Proposition 9 *[AY99] To know whether a CFM or a MSC-graph satisfies an LTL formula is undecidable.*

To know whether a universally b-bounded CFM $\mathcal{A}$ satisfies an LTL formula φ is PSPACE in $|\varphi|\wp$ and linear in $b + |\mathcal{A}|$.

To know whether a regular CMSC-graph G satisfies an LTL formula φ is PSPACE in $|\varphi||G|$.

Proposition 10 *[MM01] To know whether a CFM satisfies an MS0 formula on MSCs is undecidable.*

To know whether a safe CMSC-graph or an existentially b-bounded CFM satisfies an MSC formula φ is non elementary in $|\varphi|$.

[1] Happy is he who gets to know the reasons for things.

4.2 MSC-Graphs

4.2.1 Membership

Proposition 11 *To know whether an MSC belongs to the language of a CFM can be tested in linear time.*

To know whether an MSC M belongs to the language of a CMSC-graph is NP-complete in the number $\wp$ of processes, and NLOGSPACE in $|M|$ [AEY01].

Proposition 12 *[GMSZ02] To know whether an MSC belongs to the language of a local-choice CMSC-graph is NLOGSPACE-complete and in quadratic time.*

4.2.2 Model-Checking

Applying theorem 4 on page 23 and [MP99], we obtain:

Theorem 8 *To know whether an MSC-graph accepts every MSC is undecidable [MP99].*

Intersection of two MSC-graphs is undecidable [MP99].

Let G be a safe CMSC-graph, H be a globally cooperative CMSC-graph and $\mathcal{A}$ be a CFM. Then we can decide the following questions [GKM04]:

- $\mathcal{L}(G) \cap \mathcal{L}(H) = \emptyset$ *is PSPACE-complete in* $|H|, b_G$.
- $\mathcal{L}(G) \subseteq \mathcal{L}(H)$ *is EXPSPACE-complet in* $|H|, b_G$.
- $\mathcal{L}(G) \cap \mathcal{L}(\mathcal{A}) = \emptyset$ *is PSPACE in the number* $\wp$ *of processes.*
- $\mathcal{L}(G) \subseteq \mathcal{L}(\mathcal{A})$ *is EXPSPACE in the number of processes* $\wp$.

Proposition 13 *[GMSZ02] Let G be a safe CMSC-graph and H be a local-choice CMSC-graph.*

- $\mathcal{L}(G) \cap \mathcal{L}(H) = \emptyset$ *is PSPACE in* $\wp$.
- $\mathcal{L}(G) \subseteq (H)$ *is EXPSPACE in* $\wp$.

4.2.3 Local-Choice MSC-Graphs

Theorem 9 *[GMSZ02] Let G and G' be two local-choice MSC-graphs.*

1. *To decide whether $\mathcal{L}(G) \cap \mathcal{L}(G') \neq \emptyset$ is NLOGSPACE-complete. It can be solved in time $O(\wp^2(|G| + |G'|)^2)$.*

2. *To decide whether $\mathcal{L}(G) \subseteq \mathcal{L}(G')$ is PSPACE-complete in $|G'|$.*

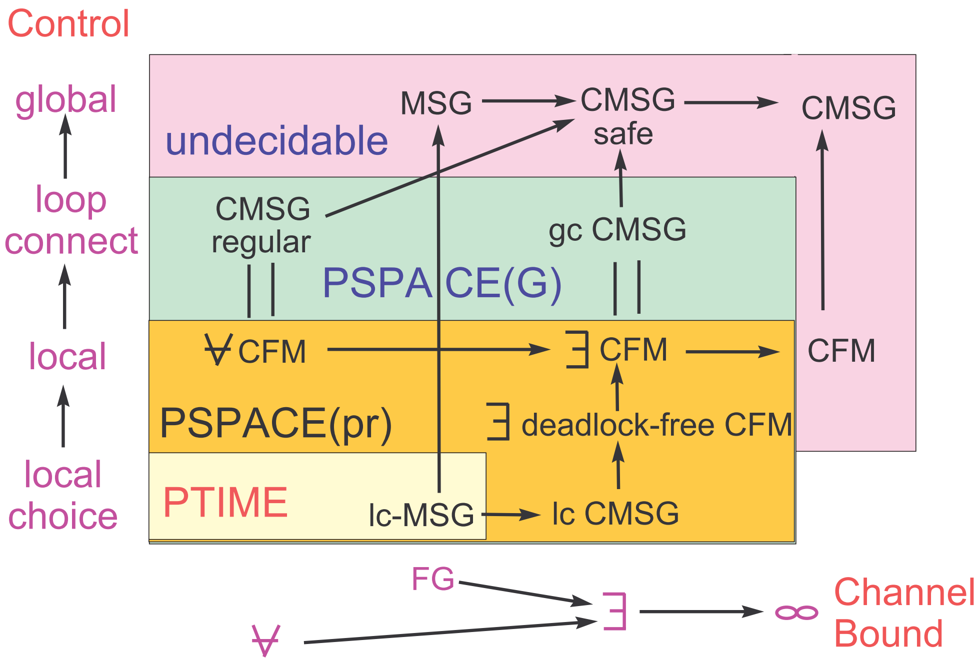


Figure 4.1: Complexity of the emptiness of the intersection of two (C)MSC-graphs.

4.3 Template MSC

We define Template MSCs [GMMP04].

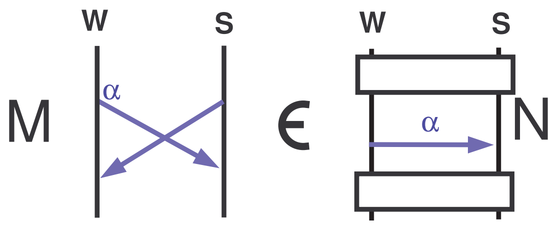


Figure 4.2: A Template MSC N representing the set of CMSC containing the message α.

4.3.1 Expressivity

Proposition 14 *[GMMP04]*
Let b be a positive integer and Φ be a template formula.

1. *To know whether there exists an MSC $\exists$-b-bounded in $\mathcal{L}(\Phi)$ is decidable.*

2. *It is undecidable to know whether $\mathcal{L}(\Phi) \neq \emptyset$.*

4.3.2 Model-Checking

Proposition 15 *[GMMP04] To know whether a CFM satisfies a template formula is undecidable*

We give here a table summing up the complexity of model-checking of a safe CMSC-graphs against a template formulas φ [GMMP04].

$S \models \varphi$	Assume	Guarantee
PTIME	closed gaps and S complete or one closed gap	$\pm$ one closed gap, no disjunction
PSPACE	no restriction	negative templates or $\pm$ two gaps
EXPSPACE	no restriction	no restriction

Chapter 5

Heuristics for Reducing the Size of the Models

Hmmmm, donuts!
Homer

This chapter gathers the results from the paper "High-Level Message Sequence Charts and Projections" with Loïc Hélouët and Anca Muscholl [GHM03], and the results from "Pattern Matching and Membership for Hierarchical Message Sequence Charts" with Anca Muscholl [GM02].

5.1 Projections

We define the projection of MSCs and of CMSC-graphs [GHM03].

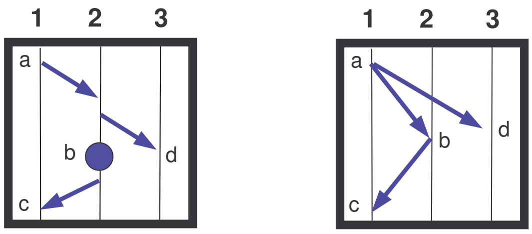


Figure 5.1: a) an MSC b) projection on $\{a, b, c, d\}$

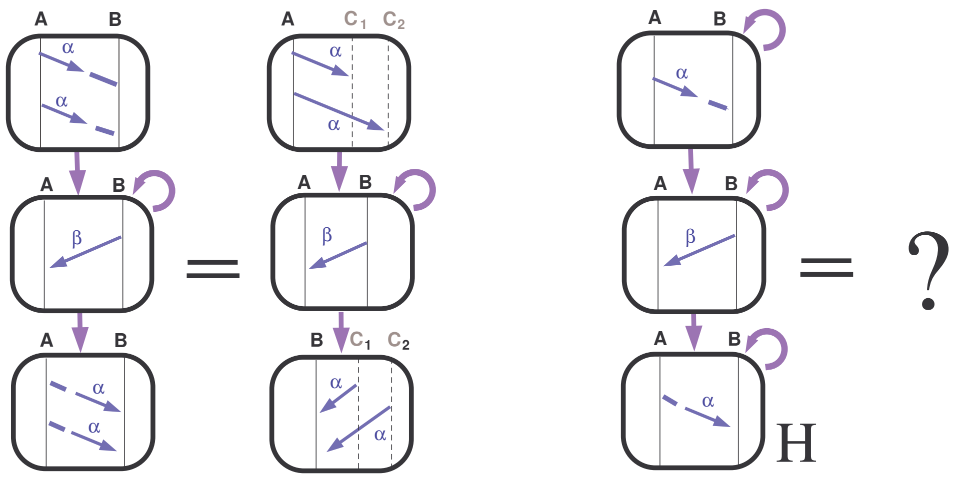


Figure 5.2: An MSC-graph Projection and a CMSC-graph.

5.1.1 Projections et Safeness

Theorem 10 *[GHM03] Projections of MSC-graphs (or safe CMSC-graphs) and safe CMSC-graphs have the same expressivity.*

5.1.2 Characterization of MSC-graphs

We define the set Gen(G) of atoms of a CMSC-graph G [HM00, HMNKT00, GHM03].

Proposition 16 *[GHM03] A safe CMSC-graph G is equivalent to some MSC-graph if and only if its set of atoms Gen(G) is finite.*

5.1.3 Atoms are Regular

Proposition 17 *[GHM03] Let G be a safe CMSC-graph. Then one can build an automaton accepting representatives for Gen(G).*

Theorem 11 *[GHM03] We can test in PSPACE (co-NP-hard) whether a safe CMSC-graph G is equivalent to some MSC-graph. If the test is positive, then one can build an equivalent MSC-graph at most doubly exponential in the size of G.*

Hence, we can depict in figure 5.3 a diagram expressing the decidable test between classes.

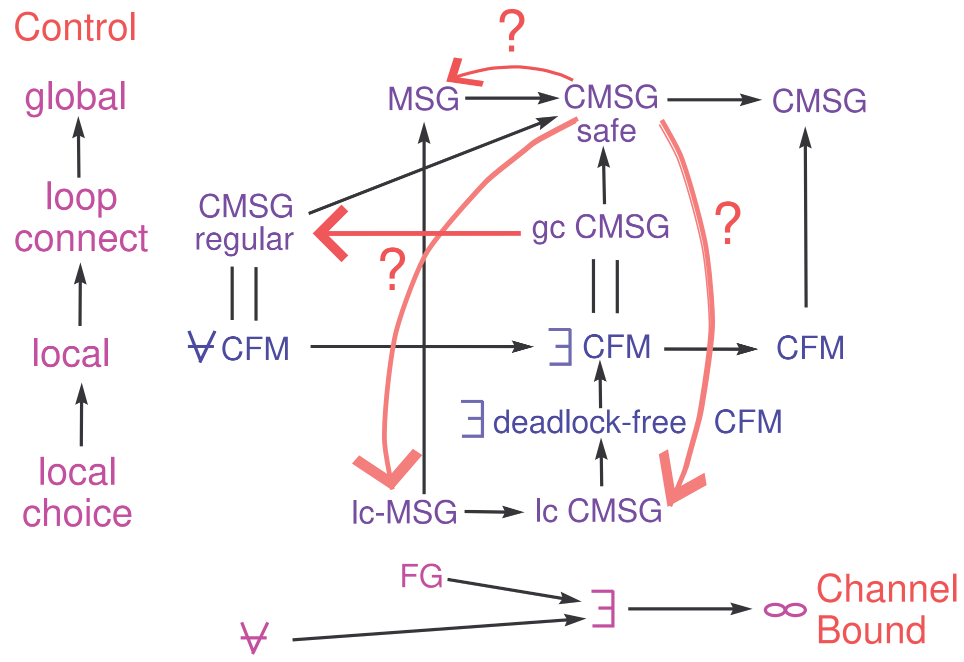


Figure 5.3: Decidable Tests.

5.2 Compression and Hierarchy

5.2.1 Syntax and Semantics of nested MSC.

We define nested MSCs and MSC-graphs (nMSC) [itu96, GM02].

5.2.2 Pattern Matching

Theorem 12 *[GM02] Let M, N be two nMSCs. To know whether M is a pattern of N can be solved in time $O(\wp^2|M|^2|N|^2)$.*

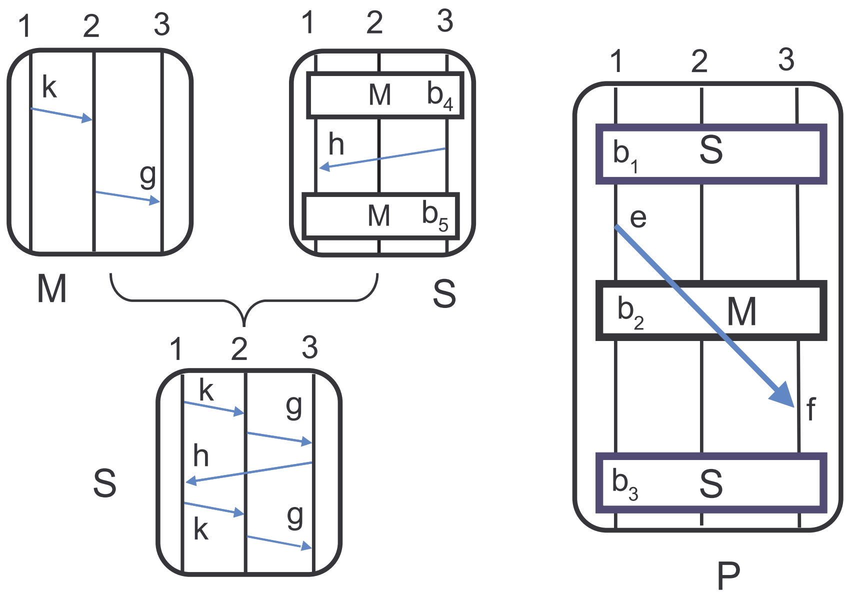


Figure 5.4: An nMSC P using two references S and M.

5.2.3 Membership Problems

Theorem 13 *Let W be a nested word and $\mathcal{A}$ be a hierarchical automaton. The membership problem $W \in L(\mathcal{A})$ is PSPACE-complete [GM02].*

If the alphabet is unary, then the membership problem is NP-complete [Ryt99].

M	G	word	MSC
Flat	Nested	P	NP-complet
Flat	Nested	P	NP-complet
Nested	Flat	P	PSPACE-complet
Nested	Nested	PSPACE-complet	PSPACE-complet

Figure 5.5: Complexity of the membership problem.

Bibliography

[ACE+03] Rajeev Alur, Swarat Chaudhuri, Kousha Etessami, Sudipto Guha, and Mihalis Yannakakis. Compression of partially ordered strings. In *CONCUR 2003 - Concurrency Theory, 14th International Conference, Marseille, France, September 3-5, 2003, Proceedings*, volume 2761 of *Lecture Notes in Computer Science*, pages 42–56. Springer, 2003.

[AEY00] Rajeev Alur, Kousha Etessami, and Mihalis Yannakakis. Inference of message sequence charts. In *Proceedings of the 22nd International Conference on Software Engineering, Limerick (Ireland)*, pages 304–313. ACM, 2000.

—— A characterization is given for finite sets of MSCs which are implementable into CFM (without control data), with or without deadlocks.

[AEY01] Rajeev Alur, Kousha Etessami, and Mihalis Yannakakis. Realizability and verification of MSC graphs. In *Proceedings of the 28th International colloquium on Automata, Languages and Programming (ICALP'01), Crete (Greece) 2001*, number 2076 in Lecture Notes in Computer Science, pages 797–808. Springer, 2001.

—— This paper considers the problem of implementing an MSC-Graph without adding control data. This implementation gives a communicating finite state machine, either with and without deadlocks. It generalizes [AEY00].

[AHP96] Rajeev Alur, Gerard H. Holzmann, and Doron A. Peled. An analyzer for message sequence charts. *Software Concepts and Tools*, 17(2):70–77, 1996.

[AKY99] Rajeev Alur, Sampath Kannan, and Mihalis Yannakakis. Communicating hierarchical state machines. In *Automata, Languages and Programming, 26th International Colloquium, ICALP'99, Prague, Czech Republic*, number 1644 in Lecture Notes in Computer Science, pages 169–178. Springer, 1999.

—— An extension of [AY98] considering automata able to communicate through FIFO message queues.

[AY98] Rajeev Alur and Mihalis Yannakakis. Model checking of hierarchical state machines. In ACM, editor, *SIGSOFT '98, Proceedings of the ACM SIGSOFT International Symposium on Foundations of Software Engineering, Lake Buena Vista, Florida, USA*, pages 175–188, 1998.

—— The authors consider a model of automaton where (SLP) compression is allowed. In other words, one can reuse macros already defined. They show that the model checking against an LTL formula is still polynomial in the size of the compressed automaton.

[AY99] Rajeev Alur and Mihalis Yannakakis. Model checking of message sequence charts. In *Proceedings of the 10th International Conference on Concurrency Theory CONCUR'99, Eindhoven (The Netherlands)*, number 1664 in Lecture Notes in Computer Science, pages 114–129. Springer, 1999.

—— Same result as [MP99] with the same class of regular MSC-Graph considered (called bounded MSC-graph here).

[BBF+01] B. Berard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, and P. Schnoebelen. *Systems and Software Verification. Model-Checking Techniques and Tools*. Springer, 2001.

—— The LSV book for Model-Checking.

[BJR97] G. Booch, I. Jacobson, and J. Rumbaugh. *Unified Modeling Language User Guide*. Addison-Wesley, 1997.

[BL04] Benedikt Bollig and Martin Leucker. Message-passing automata are expressively equivalent to EMSO logic. In *Proceedings of the 15th International Conference on Concurrency Theory (CONCUR'04)*, number 3170 in Lecture Notes in Computer Science, pages 146–160, London, UK, 2004. Springer.

—— Here, EMSO means a weaker version of EMSO, where only the message order and immediate order on processes is allowed. It is unlikely that the same result holds with the (not only immediate) process order.

[BM03] Nicolas Baudru and Rémi Morin. Safe implementability of regular message sequence chart specifications. In *Proceedings of the ACIS Fourth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD'03), October 16-18, 2003, Lübeck, Germany*, pages 210–217. ACIS, 2003.

[Büc60] J.R. Büchi. On a decision method in restricted second order arithmetic. In *Proc. Internat. Congr. on Logic, Methodology and Philosophy of Science*, pages 1–11. Stanford Univ. Press, Stanford, CA, 1960.

[BZ83] D. Brand and P. Zafiropulo. On communicating finite-state machines. *Journal of the ACM*, 30(2):323–342, 1983.

—— This paper considers automata communicating through unbounded FIFO buffers. This paper does not show the equivalence with Turing machines, which was only shown in an IBM internal report of 1982.

[CDHL00] Benoît Caillaud, Philippe Darondeau, Loïc Hélouët, and Gilles Lesventes. HMSCs as partial specifications... with PNs as completions. In *MOVEP*, 2000.

—— The authors assume that an MSC-graph specification is not complete. Hence, in order to realize it, a bigger model is allowed. Here, they consider Petri Net as the bigger model.

[CGP00] Edmund Clarke, Orna Grumberg, and Doron Peled. *Model Checking*. MIT Press, 2000.

[DG02] Volker Diekert and Paul Gastin. LTL is expressively complete for Mazurkiewicz traces. *Journal of Computer and System Sciences*, 64:396–418, 2002.

[DG04] Volker Diekert and Paul Gastin. Pure future local temporal logics are expressively complete for Mazurkiewicz traces. In *Proceedings of the 6th Latin American theoretical INformatics Symposium (LATIN'04)*, number 2976 in Lecture Notes in Computer Science, pages 232–241. Springer, 2004.

—— The paper settles a long standing problem for Mazurkiewicz traces: the pure future local temporal logic defined with the basic modalities exists-next and until is expressively complete. The analogous result with a global interpretation was solved some years ago by Thiagarajan and Walukiewicz (1997) and in its final form without any reference to past tense constants by Diekert and Gastin (2000). Each, the (previously known) global or the (new) local result generalizes Kamp's Theorem for words, because for sequences local and global viewpoints coincide. But traces are labelled partial orders and then the difference between an interpretation globally over cuts (configurations) or locally at points (events) is significant. For global temporal logics the satisfiability problem is non-elementary (Walukiewicz 1998), whereas for local temporal logics both the satisfiability problem and the model checking problem are solvable in PSPACE (Gastin and Kuske 2003) as in the case of words. This makes local temporal logics much more attractive.

[DR95] V. Diekert and G. Rozenberg. *Book of Traces*. World Scientific, Singapore, 1995.

[Gen04] Blaise Genest. Compositionality everywhere, the end of MSC-graphs?, 2004.

—— Internal LIAFA report, prior to the paper [GKM04].

[GHM03] Blaise Genest, Loïc Hélouët, and Anca Muscholl. High-level message sequence charts and projections. In *Proceedings of the CONCUR 2003: Marseilles, France*, number 2761 in Lecture Notes in Computer Science, pages 308–322. Springer, 2003.

—— This paper considers abstraction for MSC-graphs, by means of projection of events. More formally, one can delete some sends without deleting the corresponding receives (or vice versa), hence yielding a complicated partial-order relation. It does not always corresponds to an MSC-graph, but it corresponds to a safe CMSC-graph. An algorithm is given to reconstruct an MSC-graph when it is possible, together with model checking against an MSC-graph.

[GJ78] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. Freeman, San Francisco, 1978.

[GK03] P. Gastin and D. Kuske. Satisfiability and model checking for MSO-definable temporal logics are in PSPACE. In *Proceedings of the 14th International Conference on Concurrency Theory (CONCUR'03)*, number 2761 in Lecture Notes in Computer Science, pages 222–236. Springer, 2003.

[GKM04] Blaise Genest, Dietrich Kuske, and Anca Muscholl. A kleene theorem for a class of communicating automata with effective algorithms, 2004.

—— Work in progress, follow up of [Gen04].The restriction considered here is an existential bound on the channel buffer. With that restriction, it is shown that CFMs, globally cooperative CMSC-graphs and MSO are expressively equivalent.

[GM02] Blaise Genest and Anca Muscholl. Pattern matching and membership for hierarchical message sequence charts. In *Proceedings of the LATIN 2002: Theoretical Informatics, 5th Latin American Symposium, Cancun, Mexico*, number 2286 in Lecture Notes in Computer Science, pages 326–340. Springer, 2002.

—— The only paper considering hierarchy for MSCs/MSC-graphs (HMSCs for hierarchical MSCs). Using hierarchy by means of macro can be seen as using (SLP) compression. pattern matching and membership are considered.

[GM03] Blaise Genest and Anca Muscholl. Don't choose globally, implement easily, 2003.

—— Internal report of LIAFA, giving a characterization for local choice MSC-graphs.

[GMMP04] Blaise Genest, Marius Minea, Anca Muscholl, and Doron Peled. Specifying and verifying partial order properties using templates MSCs. In *Proceedings of the Fossacs 2004, Barcelona , Spain*, number 2987 in Lecture Notes in Computer Science, pages 195–210. Springer, 2004.

—— This paper introduces a weak form of global temporal logic on MSCs using explicit gaps. Model Checking against an MSC-graph and satisfiability are considered.

[GMP01] Elsa Gunter, Anca Muscholl, and Doron Peled. Compositional message sequence charts. In *Proceedings of the Tools and Algorithms for the Construction and Analysis of Systems, 7th International Workshop (TACAS'01)*, number 2031 in Lecture Notes in Computer Science, pages 496–511. Springer, 2001. Journal version to appear in the International Journal on Software Tools for Technology Transfer.

—— This paper introduces the Compositional MSCs/MSC-graphs (CMSC/CMSC-Graph), together with safe CMSC-graph (they call them realizable CMSC-graphs, which is a really nasty name). These notions proved to be really strong, and were reused for restriction/generalization later.

[GMP04] Blaise Genest, Anca Muscholl, and Doron Peled. Message sequence charts. In *Lectures on Concurrency and Petri Nets*, volume 3098 of *Lecture Notes in Computer Science*, pages 537–558. Springer, 2004.

—— This survey gathers results contained into [MP99, GMP01, GMSZ02].

[GMSZ02] Blaise Genest, Anca Muscholl, Helmut Seidl, and Marc Zeitoun. Infinite-state high-level MSCs: Model-checking and realizability. In *Proceedings of the 29th International colloquium on Automata, Languages and Programming (ICALP'02), Malaga (Spain), 2002*, number 2380 in Lecture Notes in Computer Science, pages 657–668. Springer, 2002. journal version to appear in JCSS.

—— This paper introduces a notion generalizing regular MSC-graph, namely global-cooperativity (gc-MSC-graphs). Subclasses of gc-MSC-graphs are also studied, namely local-choice and local-cooperativity. These three classes impose no universal bound on the message channels, hence are not regular. The complexity for model checking is given, together with a deadlock free implementation of any local-choice MSC-graphs.

[HJ00] Loïc Hélouët and Claude Jard. Conditions for synthesis of communicating automata from HMSCs. In *5th International Workshop on Formal Methods for Industrial Critical Systems*, Berlin, 2000.

—— This paper focuses on giving sufficient conditions to be able to implement an MSC-graph. They define local-choice MSC-Graph, and show that it is not sufficient for implementability. The reason is that control data is not allowed in the CFM implementation.

[HM00] Loïc Hélouët and Pierre Le Maigat. Decomposition of Message Sequence Charts. In *Proceedings of the 2nd Workshop on SDL and MSC (SAM2000), Col de Porte, Grenoble*, pages 46–60, 2000.

—— One of the first definition of atoms. Moreover, a nice algorithm for checking atomicity is given by testing the strong connexion of a graph (connection graph), which was reused later.

[HMNK+04] Jesper G. Henriksen, Madhavan Mukund, K. Narayan Kumar, Milind Sohoni, and P.S. Thiagarajan. A Theory of Regular MSC Languages. *To Appear in : Information and Computation.*, pages –, 2004.

—— Survey paper gathering works done in [HMNKT00, **?**, **?**]. More precisely, it defines regular MSC languages, CFM languages, MSO-definable language and shows that their restrictions to universally bounded languages are equivalent. Moreover, regular MSG languages equals their restrictions on b-bounded and finitely generated languages.

[HMNKT00] Jesper G. Henriksen, Madhavan Mukund, K. Narayan Kumar, and P.S. Thiagarajan. On message sequence graphs and finitely generated regular MSC languages. In *Proceedings of the 27th International colloquium on Automata, Languages and Programming (ICALP'00), Geneva (Switzerland), 2000*, number 1853 in Lecture Notes in Computer Science, pages 675–686. Springer, 2000.

—— The authors show that regular MSC-graphs are equivalent to finitely generated regular sets of MSCs. Moreover, it is undecidable to check whether an MSC-Graph is equivalent to some regular MSC-Graph. One of the first definition of atoms.

[HU79] J. E. H. Hopcroft and Jeffrey D. Ullman. *Introduction to Automate Theory, Languages, and Computation*. Reading, Menlo Park ,London. Addison-Wesley Publishing Company, 1979.

[itu96] Itu-ts. itu-ts recommendation z.120: Message sequence chart (msc). itu-ts, geneva, september 1996., 1996.

—— The standard for MSCs and HMSCs.

[Kle56] Stephen Kleene. Representation of events in nerve nets and finite automata. In Shannon and McCarthy, editors, *Automata Studies*, pages 3–42. Princeton University Press, Princeton, N.J., 1956.

[Kus03] Dietrich Kuske. Regular sets of infinite message sequence charts. *Information and Computation*, 187:80–109, 2003.

—— This paper features an encoding of universally bounded communication with traces.

[LL90] Leslie Lamport and Nancy Lynch. Distributed computing: models and methods. In *Handbook of Theoretical Computer Science*, volume B, pages 1157–1200. Elsevier, Amsterdam, 1990.

[LM04] Markus Lohrey and Anca Muscholl. Bounded MSC communication. *Information and Computation*, 189:135–263, 2004.

—— Definition of the universal and existential bounds, as well as algorithms to calculate the bounds.

[Lyn96] Nancy Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers, 1996.

[MM01] P. Madhusudan and B. Meenakshi. Beyond message sequence graphs. In *Proceedings of the FST TCS 2001: Foundations of Software Technology and Theoretical Computer Science, 21st Conference, Bangalore (India)*, number 2245 in Lecture Notes in Computer Science, pages 256–267. Springer, 2001.

—— The Authors show that one can model-check MSO against safe CMSG (they define the term 'CMSG' with the definition of safe CMSG). The basic idea is that safe CMSGs give an existential bound b for representatives. One can construct an automaton closed under commutation recognizing b-bounded executions which satisfy the MSO formula. It generalizes [**?**].

[Mor02] Rémi Morin. Recognizability vs realizability and mso-definability in unbounded high level message sequence charts. In *Proceedings of the 19th Symposium on Theoretical Aspects of Computer Science (STACS'02, Antibes, France)*, number 2285 in Lecture Notes in Computer Science, pages 523–534. Springer, 2002.

—— An algebraic look into MSC languages. More precisely, it compares recognizability, regularity, rationality and MSO-definability of infinite MSC languages, extending results of [HMNK^{+}04]. One result is that one can check whether a gc-MSC-Graph (c-HMSC) is implementable into a non-FIFO CFM.

[MP99] Anca Muscholl and Doron Peled. Message sequence graphs and decision problems on Mazurkiewicz traces. In *Proceedings of the 24thSymposium on Mathematical Foundations of Computer Science (MFCS'99), Szklarska Poreba (Poland) 1999*, number 1672 in Lecture Notes in Computer Science, pages 81–91. Springer, 1999.

—— Definition of strong loop-connectedness, which led to the first decidability result for Model Checking between MSC-Graphs.

[MR97] S. Mauw and M.A. Reniers. High-level message sequence charts. In *SDL'97: Time for Testing - SDL, MSC and Trends. Proceedings of the 8th SDL Forum, Evry (France) 1997*, pages 291–306, 1997.

—— Definition of the HMSCs.

[Och85] Edward Ochmański. Regular behaviour of concurrent systems. *Bulletin of EATCS*, 27:56–67, 1985.

[Pap94] C.H Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.

—— A well known book.

[Pel00] Doron Peled. Specification and verification of message sequence charts. In *Proceedings of Formal Techniques for Distributed System Development, FORTE/PSTV 2000, Pisa, Italy*, pages 139–154, 2000.

—— Survey about Model-Checking against Partial Order Logic. Idea that if the Logic is closed under commutation, then we can choose representatives for the MSG.

[Ryt99] Wojciech Rytter. Algorithms on compressed strings and arrays. In *SOFSEM '99, Theory and Practice of Informatics, 26th Conference on Current Trends in Theory and Practice of Informatics, Milovy, Czech Republic*, number 1725 in Lecture Notes in Computer Science, pages 48–65. Springer, 1999.

—— Rytter uses the idea of Plandowsky [**?**]. That is, one can check in PTIME the equality of two words compressed by SLPs (close to gzip compressions). Rytter extends this result to pattern matching of two such words, plus membership for a compressed word in an uncompressed automaton (both in PTIME).

[SC02] Bikram Sengupta and Rance Cleaveland. Triggered Message Sequence Charts. In *SIGSOFT 2002/FSE-10*. ACM Press, 2002.

—— Definition of Triggered MSCs, example of WSR system. There is a nice idea of describing system by (parallel) composition of several local patterns, then restricting the global behavior. Anyway, the formalism is ad hoc for implementation, hence weird (postcondition before precondition). It is so because the semantic is per process instead than the MSC composition semantic.

[TW02] P.S. Thiagarajan and Igor Walukiewicz. An expressively complete linear time temporal logic. *Information and Computation*, 179(2):230–249, 2002.

[usb96] Usb 1.1 specification, 1996. ——— available at `http://www.usb.org/developers/docs/usbspec.zip`.

[Wal98] Igor Walukiewicz. Difficult Configurations - On the Complexity of LTrL. In *Proceedings of the 25th International colloquium on Automata, Languages and Programming (ICALP'98), 1998*, number 1443 in Lecture Notes in Computer Science, pages 140–151. Springer, 1998.

——— First paper where the idea of counters for Turing machine appears. It is used to encode exponential space TM with polynomial memory.

[Zie87] Wieslaw Zielonka. Note on finite asynchronous automata R.A.I.R.O. *Informatique Théorique et Applications*, 21:99–135, 1987.