Message Sequence Charts: A Survey

Blaise Genest
Department of Computer Science, University of Warwick
Coventry, CV4 7AL, United Kingdom

Anca Muscholl
LIAFA, University of Paris 7
2, pl. Jussieu, F-75251 Paris

Abstract

Message sequence charts *(MSC) are a graphical notation standardized by the ITU and used for the description of communication scenarios between asynchronous processes. This talk concerns the formal analysis of MSC-based specifications in relation with communicating finite-state machines. We discuss two basic validation problems about MSCs specifications, model-checking and implementability.*

1. Introduction

Modeling and validation, whether formal or ad-hoc, are important steps in system design. Over the last couple of decades, various methods and tools were developed for decreasing the amount of design and development errors. A common component of such methods and tools is the use of *formalisms* for *specifying* the behavior and requirements of the system. Experience has shown that some formalisms, such as finite-state machines, are particularly appealing, due to their convenient mathematical properties. In particular, the expressive power of finite-state machines is identical to *regular languages*, an important and well-studied class of languages. Although their expressiveness is restricted, finite-state machines are used for the increasingly successful automatic verification of software and hardware, also called *model-checking* [7, 8]. One of the biggest challenges in developing new validation technology based on finite-state machines is to make this model popular among system engineers.

The *Message Sequence Charts* (MSC) model has become popular in software development throughout its visual representation, depicting the involved processes as vertical lines, and each message as an arrow between the source and the target processes, according to their occurrence order. An international standard [1], and its inclusion in the UML standard, has increased the popularity. The standard has also extended the notation to *Message Sequence Graphs* (MSGs), which consist of finite transition systems, where each state embeds a single MSC. Encouraged by the success of the formalism among software developers, techniques and tools for analyzing MSCs and MSGs have been developed.

In this talk we describe the formal analysis of MSCs and MSGs. The class of systems that can be described using this formalism does not directly correspond to a well-studied class such as regular languages. It turns out that MSGs are incomparable with the class of finite-state communication protocols. One thus needs to separately study the expressiveness of MSG languages, and adapt the validation algorithms. Several new algorithms are suggested in order to check MSG properties, mostly related to an automatic translation from MSG specification into skeletons of concurrent programs. Our survey concentrates on the following subjects:

Expressiveness: Comparing the expressive power of MSGs to the expressive power of other formalisms, in particular communicating finite-state machines or logics [10, 11, 20, 21, 22, 28].

Verification: The ability to apply automatic verification algorithms on MSGs, and the various formalisms used to define properties of MSGs [5, 6, 13, 19, 25, 26, 31].

Implementability: The ability to obtain an automatic translation from MSG specification into a machine-oriented model (such as communicating finite-state machines) that can be easily converted into skeletons of code [10, 14, 17, 18, 23].

Generalizations and restrictions: Various extensions and restrictions of the standard notation are suggested in order to capture further systems, and on the other hand, to obtain decidability of important decision procedures. Among these models we have *Live Sequence Charts* [16], *Triggered MSCs* [33], *Dynamic Message Sequence Charts* [24], *Netcharts* [29] and *Template MSCs* [12]. The motivation behind these models is to increase the expressiveness of the notation, and to make their usage by designers even more convenient.

Proceedings of the Fifth International Conference on Application of Concurrency to System Design (ACSD'05)
1550-4808/05 $ 20.00 **IEEE**

References

[1] ITU-TS recommendation Z.120, 1996.

[2] R. Alur, K. Etessami, and M. Yannakakis. Inference of Message Sequence Charts. In *Proceedings of the 22nd International Conference on Software Engineering, Limerick (Ireland)*, pages 304–313. ACM, 2000.

[3] R. Alur, K. Etessami, and M. 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.

[4] R. Alur, G. H. Holzmann, and D. A. Peled. An analyzer for message sequence charts. *Software Concepts and Tools*, 17(2):70–77, 1996.

[5] R. Alur and M. 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.

[6] H. Ben-Abdallah and S. Leue. Syntactic detection of process divergence and non-local choice in message sequence charts. In *Proceedings of the Tools and Algorithms for the Construction and Analysis of Systems, Third International Workshop, TACAS'97*, number 1217 in Lecture Notes in Computer Science, pages 259–274, Enschede, The Netherlands, 1997. Springer.

[7] 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.

[8] E. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 2000.

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

[10] B. Genest. The odyssey of MSC-graphs. PhD thesis, Université Paris 7, 2004 (in French).

[11] B. Genest, D. Kuske and A. Muscholl. A Kleene Theorem and Model Checking for a Class of Communicating Automata. In *Proceedings of Developments in Language Theory (DLT)*, number 3340 in Lecture Notes in Computer Science, pages 30-48. Springer, 2004.

[12] B. Genest, M. Minea, A. Muscholl, and D. Peled. Specifying and verifying partial order properties using template MSCs. In *Proceedings of the 7th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'04)*, number 2987 in Lecture Notes in Computer Science, pages 195-210. Springer, 2004.

[13] B. Genest and A. 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.

[14] B. Genest, A. Muscholl, H. Seidl, and M. Zeitoun. Infinite-state High-level MSCs: Model-checking and realizability. In *Proceedings of the 29thInternational colloquium on Automata, Languages and Programming (ICALP'02), Malaga (Spain), 2002*, number 2380 in Lecture Notes in Computer Science, pages 657–668. Springer, 2002.

[15] E. Gunter, A. Muscholl, and D. 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 in *International Journal on Software Tools for Technology Transfer (STTT)* 5(1): 78-89 (2003).

[16] D. Harel and R. Marelly. Come, Let's Play: Scenario-Based Programming Using LSCs and the Play-Engine. Springer 2003.

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

[18] D. Harel and H. Kugler Synthesizing State-Based object systems from LSC Specifications. Technical Report of Weizmann Institute of Science, 1999.

[19] L. Hélouët and P. 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.

[20] J. G. Henriksen, M. Mukund, K. Narayan Kumar, and P. 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.

[21] J. G. Henriksen, M. Mukund, K. Narayan Kumar, and P. Thiagarajan. Regular collections of Message Sequence Charts. In *Proceedings of the 25th Symposium on Mathematical Foundations of Computer Science (MFCS'00), Bratislava (Slovakia), 2000*, number 1893 in Lecture Notes in Computer Science, pages 405–414. Springer, 2000.

[22] D. Kuske. A Further Step towards a Theory of Regular MSC Languages. In *Proceedings of the 19th Annual Symposium on Theoretical Aspects of Computer Science (STACS'02), Juan-les-Pins (France), 2002*, number 2285 in Lecture Notes in Computer Science, pages 489–500. Springer, 2002. Journal version in *Inf. Comput* 187(1): 80-109 (2003)

[23] M. Lohrey. Safe Realizability of High-level Message Sequence Charts. In *Proceedings of the Concurrency Theory, 13th International Conference (CONCUR'02)*, number 2421 in Lecture Notes in Computer Science, pages 177–192. Springer, 2002.

[24] M. Leucker, P. Madhusudan and S. Mukhopadhyay. Dynamic Message Sequence Charts. In *FSTTCS'02*, LNCS 2556, pp. 253-264, 2002.

[25] M. Lohrey and A. Muscholl. Bounded MSC communication. *Information and Computation*, (189):135–263, 2004.

[26] P. Madhusudan. Reasoning about sequential and branching behaviours of Message Sequence 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 809–820. Springer, 2001.

[27] P. Madhusudan and B. Meenakshi. Beyond Message Sequence Graphs. In *FSTTCS'01*, LNCS 2245, pp. 256-267, 2001.

[28] R. Morin. Recognizable Sets of Message Sequence Charts In *Proceedings of the 19th Annual Symposium on Theoretical Aspects of Computer Science (STACS'02), Juan-les-Pins (France), 2002*, number 2285 in Lecture Notes in Computer Science, pages 523–540. Springer, 2002.

[29] M. Mukund, K. Narayan Kumar, and P.S. Thiagarajan. Netcharts: bridging the gap between HMSCs and executable specifications. In In *Proceedings of the Concurrency Theory, 14th International Conference (CONCUR'03)*, number 2761 in Lecture Notes in Computer Science, pages 296–310. Springer, 2003.

[30] A. Muscholl and D. Peled. Message sequence graphs and decision problems on Mazurkiewicz traces. In *Proceedings of the 24th Symposium 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.

[31] A. Muscholl, D. Peled, and Z. Su. Deciding properties of Message Sequence Charts. In *Proceedings of the 1st International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'98), Lisbon, Portugal, 1998*, number 1378 in Lecture Notes in Computer Science, pages 226–242. Springer, 1998.

[32] D. 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.

[33] B. Sengupta and R. Cleaveland. Triggered Message Sequence Charts. In *Proceedings of SIGSOFT 2002/FSE-10*, pages 167–176, ACM Press, 2002.