Decidability Issues for Decentralized Controllability of Open Nets

Karsten Wolf

Universität Rostock, Institut für Informatik

Abstract. We sketch an undecidability result concerning the decentralized controllability problem of open nets and discuss some consequences.

1 Introduction

During the last years, we have used open nets (Petri nets with distinguished interface places) as a formal model for the behavior (protocol) of services [Wol09b]. Open nets can also be used as a tool in divide-and-conquer approaches to Petri net verification [Zai06,OWW10]. For this class of nets, it is interesting to study the interaction with their environment [Wol09a]. To this end, various notions of *controllability* [RW87] are in the main focus. So far, we concentrated on *centralized controllability* which asks whether there is a (monolithic) environment that can be composed in such a way to the given open net that the overall system is deadlock free or deadlock free and livelock free. In both cases, we were able to come up with decision procedures [LW10a] in the case where the given net has finitely many states while we showed undecidability for the case that the given open net is unconstrained [MSSW08].

Decentralized controllability is applied to an open net that has a partitioned interface. The question is: is there a tuple of environments, each communicating with one part of the interface and not directly communication with each other, such that the overall system is deadlock free or deadlock and livelock free? In [Wol09a], we gave a procedure only for the case that N does not run into cycles, i.e. never visits states twice. Decentralized controllability is a very useful notion as several practically relevant problems can be reduced to it:

- *Adaptibility* [GMW10]: Given a service S and a specification of elementary activities that specify the general space of actions of any adapter, is there another service R such that S and R can be made compatible by a mediating adapter? In this setting, the service R and the control unit of the adapter form a decentralized controller of a system consisting of S and the implementation of the possible elementary activities of the adapter.
- *Realizability* [LW10b]: Given a choreography (i.e. a language of event sequences to happen on the wires), are there services that interact in such a way that the observed communication fits to that language? In this case, the choreography can be transformed into an open net which has the tuple of realizing services as controller.

The notion of decentralized controllability must not be mixed up with the one of *autonomous controllability*. In the latter notion, the question is whether a decentralized

controller can be found in such a way that the different parts are not even coordinated at build time.

In this paper, we consider decentralized controllability for the setting where we want to enforce deadlock freedom and livelock freedom on the overall system. We further rule out cotrollers where, at any stage of communication, more than a given number k of tokens is pending on a message channel and we rule out unbounded open nets as these would be covered by the undecidability result of [MSSW08]. These restrictions ensure that the overall system is finite state and the undecidability result of [MSSW08] does not cover the setting.

To our best knowledge, undecidability of the mentioned problem has not been shown before while similar problem have been studied. Tripakis shows [Tri04] undecidability of decentralized controllabilty in a different setting. For his problem, the actual correctness property to be enforced by the synthesized controller is a parameter that is part of the input (in the shape of a regular language of events) while we consider the fixed setting of deadlock and livelock freedom. His undecidability result relies on coding Post's Correspondance Problem (PCP) into the mentioned language. Bontemps and Shoppens show [BS07] undecidability of the distributed realizability of a Life Sequence Chart (LSC) specification. Peculiarly, their setting permits communication between the distributed agents to be synthesized. In our situation, permission of direct communication between the distributed parts of the environment would immediately permit the conclusion that an open net is decentralized controllable if and only if it is centralied controllable. As we know that centralized controllability is decidable in our case, we conclude thet the setting of [BS07] includes some, maybe implicite, assumptions that are not compatible to our setting. In their argument, however, they as well use a reduction of PCP as their main argument.

Consequently, it is not surprising that our proof relies on a reduction from PCP as well. However, the particular execution of the reduction differs significantly. While Tripakis codes the PCP instance into the language used as a correctness criterion, Bontemps and Schoppens present the distibuted agents with a challenge generated from a nondeterministically selected PCP instance that is not solvable iff the PCP instance has a solution. We generate a controlling distributed strategy from a solution of a PCP instance and use the given open net as a verifier of that instance.

2 Open nets

In this section, we introduce the notion of open nets. The notion syntactically deviates a bit from earlier presentations. It is, however, semantically equivalent.

An open net extends a usual place/transition net $[P, T, F, m_0]$ with the following ingredients:

- A set M of *message shapes*,
- An *interface* I which is a set of *pins* partioned into ports Π_i ($I = \Pi_1 \cup \ldots \cup \Pi_n$, $\Pi_1 \cap \Pi_j = \emptyset$ for $i \neq j$); a pin is a pair $[m, d]$ where $m \in M$ and $d \in \{i, o\}$,
- A set Ω of *final markings*.
- A partial *labeling* that assigns a message shape to some of the transitions.

i and o represents incoming and outgoing messages, respectively. For a pin $[m, d]$, define the notion of a *dual pin* $\overline{[m, d]}$ by $\overline{[m, i]} = [m, o]$ and $\overline{[m, o]} = [m, i]$. For the interface of an open net, we require that, for all pins $[m, x]$, $[m, x] \in I$ implies $\overline{[m, x]} \notin I$. For a port Π_i, let the *dual port* $\overline{\Pi_i} = \{\overline{[m, x]} \mid [m, x] \in \Pi_i\}$.

A set of open nets $N_1, \ldots, N_k$ is *composable* iff, for all m and x, $[m, x] \in I_i$ and $[m, x] \in I_j$ implies $i = j$ (each connection is purely bilateral), and, for each pair of pins in the same port, their dual pins belong to the interface of the same net, or none of the dual pins belongs to any interface (ports signal communication with the same partner).

For composable open nets, the composition is built by

- building the disjoint partition of the places, transitions, and arcs of the involved net (if necessary by renaming);
- introducing additional places for all those pins where both pin and dual pin occur in some interface;
- inserting an arc from a transition t labeled m to the place inserted for m if $[m, o]$ occurs in the interface of the net containing t,
- inserting an arc from the place labeled m to a transition t labeled with m if $[m, i]$ occurs in the interface of the net containing t,
- removing the labels of all such connected transitions,
- letting the new ports be exactly those ports whose pins are not matched with dual pins in other nets,
- letting the initial marking be the disjoint partition $m_0 = m_{01} \oplus \ldots \oplus m_{0k}$, i.e. $m_0(p) = m_{0i}(p)$ for the unique i with $p \in \Pi_i$,
- letting $\Omega = \{m_1 \oplus \ldots \oplus m_k \mid m_i \in \Omega_i\}$.

We study the following *decentralized controllability problem*: Given some number k, open net N with ports $\Pi_1, \ldots, \Pi_n$, do there exist open nets $N_1, \ldots, N_n$ where each N_i has $\overline{\Pi_i}$ as its only port such that $N \oplus N_1 \oplus \ldots \oplus N_n$ is a k-bounded Petri net which is weakly terminating, i.e. from each reachable marking, a final marking is reachable?

3 Undecidability of Decentralized Controllability

In this section, we present a reduction of PCP to our setting of decentralized controllability. In PCP, we are given a set $d_1 = [u_1, v_1], \ldots, d_a = [u_a, v_a]$ of pairs of words over some fixed alphabet Σ. The problem is to decide whether there exists a finite word w over $\{d_1, \ldots, d_a\}$ such that $w_{[d_1 \leftarrow u_1, \ldots, d_a \leftarrow u_a]} = w_{[d_1 \leftarrow v_1, \ldots, d_a \leftarrow v_a]}$. It is well known that PCP is undecidable.

We prove:

Theorem 1. *If our decentralized controllability problem is decidable then PCP is decidable.*

We argue by translating an instance of PCP to an open net such that the PCP instance has a solution if and only if the open net is decentralized controllable. The idea is to design the open net such that the environment must be built from a candidate solution and leads to termination of the overall system if and only if the candidate solution is indeed a solution. Our open net is designed to have two ports, U and V. From port

U, we expect a sequence $d_{i_1} u_{i_1} d_{i_2} u_{i_2} \ldots$ followed by a concluding # while from V we expect a sequence $d_{i_1} v_{i_1} d_{i2} v_{i_2} \ldots$ followed by a concluding #.Together, the sequences should form a candidate solution for the PCP. Thus, U has input pins corresponding to $\Sigma \cup \{d_1, \ldots, d_a\} \cup \{\#\}$. Correspondingly, V has input ports corresponding to $\Sigma \cup \{d_1, \ldots, d_a\} \cup \{\#\}$ as well which need to be renamed bijectively in order to meet the syntactical constraints on pins. For each port there is a single output pin representing the acknowledgment of a received message.

The constructed open net behaves as follows: Whenever it sees more than one message in any of the two ports, it immediately moves into some deadlock or trap marking. This way, the partners are forced to send their content consecutively. Second, after having processed a message, it sends an acknowledgment to the sender. This way, the partners know when to savely send the next element of their candidate string. In the core part, the constructed net verifies the presented candidate strings. Verification includes

(1) Checking whether the candidate strings indeed correspond to a sequence of pairs of the given PCP instance;
(2) Checking whether the concatenation of the u_i is equal to the concatenation of the v_i.

A finite state system cannot check both items concurrently. The reason is that, in intermeadiate steps, a substring $u_1 \ldots u_b$ may have a different length than a corresponding substring $v_1 \ldots v_b$. The length difference may exceed any finite bound even if the substrings can be complemented to a solution of PCP. In the most popular proof of undecidability of PCP, the difference between $u_1 \ldots u_b$ and $v_1 \ldots v_b$ is used to code a comple Turing machine configuration including the state of the tape.

For this reason, the idea is to start with an internal nondeterministic decision and then to check only one of the above items. As the internal decision is not communicated to the environment, the partners can only control the open net by meeting both requirements.

If the internal decision is in favour of checking (1), the first input message on both ports is the identifier of some pair. If these identifiers are different or some other message is sent, we let the open net proceed into a deadlock. Otherwise, there is a branch depending on the pair identifier d_i and the open net matches the input on U with the sequence u_i and the input on V with v_i. This is clearly doable in a finite state fashion. If after that check there is # on both ports, the open net proceeds to a final state, otherwise it returns to the state where it expects the next path identifier.

If the decision is in favour of the second item, the open net checks whether the same letters are present in both ports. Thereby, all seen pair identifiers are ignored (i.e. acknowledged for stepping to the next input). If, at some stage, there appear # on both ports, we proceed to a final state. If a # appears on only one port, we proceed to a deadlock state. Otherwise, we continue forever.

At no time, any of the distributed partners is able to make educated guesses about the outcome of the internal decision between (1) and (2). In both cases, the individual communication is a straight alternation between inputting a letter of the candidate string and an acknowledgment. Any deviation from this procedure comes at the risk of deadlock if the open net is in the opposite mode.

4 Discussion

There is a few interesting questions that appear immediately.

First, what is the difference if we present the constructed net to a centralized environment? In the previous section, one crucial argument was that the partners cannot "sense" the outcome of the internal decision of the constructed open net. It turns out that a centralized partner *is* in fact able to sense that outcome in sufficiently many cases. This ability relies on the already mentioned difference between the lengths of substrings of the u_i and the v_i. If the open net is in mode (1), it processes its input such that it synchronises along the pair identifiers. That is, assuming that the length of the sequence of the u_i is longer than the one on the v_i, it proceeds farther into U than into V. Technically, "proceeding farther" corresponds to acknowledgements sent to U earlier than acknowledgnments to V. If the open net is in mode (2), it synchronises along letters of the alphabet, i.e., it would send acknowledgments for subsequent pairs on V before finishing acknowwledgements on U. This way, the outcome of the internal decision between (1) and (2) becomes visible to the partner at least on those PCP instances where all solutions go through intermediate steps with greatly diverging lengths between the u_i and v_i sequences. Once having recognised the mode of the open net, the partner can now react with nonstandard strategies. In mode (1) the partner can finish after having sent a complete pair even if the two sequences do not match. In mode (2), the partner canm send some matching letters on U and V even if there is no corresponding PCP pair. Hence, the constructed open net would not establish a valid PCP reduction.

Second, what about using only deadlock freedom as a correctnes criterion? Again, the used constructive ides does not work. If deadlock freedom is the only criterion, the distributed partner may choose an infinite sequence of pairs where, for all finite prefixes, the sequence of the u_i matched the corresponding initial segment of the v_i. There exist PCP instances with that property which do not have solutions. They can be built from nonterminating Turing machines using the translation used in the well known reduction of the halting problem to PCP. Although the PCP has no solution, the partners provide input forever without failing in the (1) or (2) checks. It is evident, that it is impossible to add another finite state check that identifies the nontermination of the input as this would require reasoning about the halting problem of Turing machines. We conclude that decidability of decentralized controllabillity w.r.t. deadlock freedom is still open.

Third, what about the autonomous setting? It is easy to see that the two partners are build-time coordinated. If they want to control the constructed open net, they have to agree on some PCP solution (although they do not communicate with each other while feeding it into the open net). In the autonomus setting, there is no reason to believe that there is any canonical choice for the individual partners that composes to a strategy.

Gven the importance of decentralized controllability for interesting problems like adaptability of realizability, the floor is now open for proposing approximations or solutions for subclasses of open nets. As one of the core aspects in our proof was the initial hidden choice between (1) and (2), a closer look into the disclosure of internal choices to the partners may be an interesting starting point.

References

[BS07] Yves Bontemps and Pierre-Yves Schobbens. The computational complexity of scenario-based agent verification and design. *J. Applied Logic*, 5(2):252–276, 2007.

[GMW10] Christian Gierds, Arjan J. Mooij, and Karsten Wolf. Reducing adapter synthesis to controller synthesis. *IEEE T. Services Computing*, 2010. (accepted for publication in July 2010).

[LW10a] Niels Lohmann and Daniela Weinberg. Wendy: A tool to synthesize partners for services. In Johan Lilius and Wojciech Penczek, editors, *31st International Conference on Applications and Theory of Petri Nets and Other Models of Concurrency, PETRI NETS 2010, Braga, Portugal, June 21-25, 2010, Proceedings*, volume 6128 of *Lecture Notes in Computer Science*, pages 297–307. Springer-Verlag, June 2010.

[LW10b] Niels Lohmann and Karsten Wolf. Realizability is controllability. In Cosimo Laneve and Jianwen Su, editors, *Web Services and Formal Methods, 6th International Workshop, WS-FM 2009, Bologna, Italy, September 4-5, 2009, Revised Selected Papers*, volume 6194 of *Lecture Notes in Computer Science*, pages 110–127. Springer-Verlag, September 2010. (in press).

[MSSW08] Peter Massuthe, Alexander Serebrenik, Natalia Sidorova, and Karsten Wolf. Can I find a partner? Undecidablity of partner existence for open nets. *Inf. Process. Lett.*, 108(6):374–378, November 2008.

[OWW10] Olivia Oanea, Harro Wimmel, and Karsten Wolf. New algorithms for deciding the siphon-trap property. In Johan Lilius and Wojciech Penczek, editors, *31st International Conference on Applications and Theory of Petri Nets and Other Models of Concurrency, PETRI NETS 2010, Braga, Portugal, June 21-25, 2010, Proceedings*, volume 6128 of *Lecture Notes in Computer Science*, pages 267–286. Springer-Verlag, June 2010.

[RW87] P.J. Ramadge and W.M. Wonham. Supervisory control of a class of discrete -event processes. *SIAM J. Control and Optimization*, 25(1), 1987.

[Tri04] Stavros Tripakis. Undecidable problems of decentralized observation and control on regular languages. *Inf. Process. Lett.*, 90(1):21–28, 2004.

[Wol09a] Karsten Wolf. Does my service have partners? *LNCS ToPNoC*, 5460(II):152–171, March 2009. Special Issue on Concurrency in Process-Aware Information Systems.

[Wol09b] Karsten Wolf. A theory of service behavior. In Oliver Kopp and Niels Lohmann, editors, *Proceedings of the 1st Central-European Workshop on Services and their Composition, ZEUS 2009, Stuttgart, Germany, March 2-3, 2009*, volume 438 of *CEUR Workshop Proceedings*, pages 1–7. CEUR-WS.org, March 2009.

[Zai06] D. A. Zaitsev. Compositional analysis of Petri nets. *Cybernetics and Systems Analysis, Volume 42, 1, 2006*, pages 126–136, 2006.