Chapter 49
Coordination Control of Distributed Discrete-Event Systems

Jan Komenda and Tomáš Masopust

Abstract The aim of this essay is to provide a brief introduction to the coordination control approach for distributed discrete-event systems with synchronous communication, where *synchronous communication* means synchronization of subsystems by the simultaneous occurrence of shared events.

49.1 Motivation

Supervisory control of distributed discrete-event systems with synchronous communication and a global specification is a difficult problem. Control synthesis relying on the composition of all subsystems is not feasible in general because of its exponential complexity in the number of subsystems. Hence a local control synthesis is preferred. However, controllers based only on the local control synthesis may be blocking, and, if not blocking, they may not reach the performance of the global control synthesis. Therefore a form of coordination between the subsystems is needed.

The coordination control architecture proposed in [11] is applicable in general and deals with a control synthesis for distributed systems with a global specification. Coordination control was first developed for prefix-closed languages in [10] and then further extended to partial observations in [6]. A non-prefix-closed extension is discussed in [7]. The approaches for prefix-closed languages are implemented in the software library libFAUDES [14].

For simplicity, the theoretical development considers the special case of two subsystems. However, the extension to more subsystems is straightforward as demonstrated in the example consisting of three subsystems.

Jan Komenda · Tomáš Masopust
Institute of Mathematics, Academy of Sciences of the Czech Republic, Žižkova 22, 616 62 Brno, Czech Republic, e-mail: komenda@ipm.cz, masopust@math.cas.cz

49.2 Problem

Consider a system given by a composition of generators G_1 and G_2 over the event sets Σ_1 and Σ_2, respectively. Let G_k be a coordinator over an event set Σ_k such that $\Sigma_k \supseteq \Sigma_1 \cap \Sigma_2$. Assume that the specification $K \subseteq L_m(G_1 \| G_2 \| G_k)$ and its prefix-closure $\overline{K}$ are conditionally decomposable with respect to event sets Σ_1, Σ_2, and Σ_k (see Section 49.3). The aim of the coordination control synthesis is to determine nonblocking supervisors S_1, S_2, S_k for respective generators such that

$$L_m(S_k/G_k) \subseteq P_k(K) \qquad \text{and} \qquad L_m(S_i/[G_i \parallel (S_k/G_k)]) \subseteq P_{i+k}(K)\,,$$

for $i = 1,2$, and the closed-loop system with the coordinator satisfies

$$L_m(S_1/[G_1 \parallel (S_k/G_k)]) \ \parallel \ L_m(S_2/[G_2 \parallel (S_k/G_k)]) = K\,. \qquad \diamond$$

Note that one could expect that the equality $L(S_1/[G_1 \parallel (S_k/G_k)]) \parallel L(S_2/[G_2 \parallel (S_k/G_k)]) = \overline{K}$ for prefix-closed languages should also be required in the statement of the problem, but it is sufficient to require the equality for marked languages since it implies that

$$\begin{aligned}\overline{K} &= \overline{L_m(S_1/[G_1 \parallel (S_k/G_k)]) \parallel L_m(S_2/[G_2 \parallel (S_k/G_k)])}\\ &\subseteq \overline{L_m(S_1/[G_1 \parallel (S_k/G_k)])} \parallel \overline{L_m(S_2/[G_2 \parallel (S_k/G_k)])} \subseteq \overline{P_{1+k}(K)} \parallel \overline{P_{2+k}(K)} = \overline{K}\,.\end{aligned}$$

If such supervisors exist, their synchronous product is a nonblocking supervisor for the global plant, cf. [5].

Example 49.1. Database transactions are examples of discrete-event systems that should be controlled to avoid incorrect behaviors. Transactions are modeled by a sequence of request (r), access (a), and exit (e) operations. Often, several users access the database, which can lead to inconsistencies when executed concurrently, because not all interleavings of operations give a correct behavior. Consider three users with events r_i, a_i, e_i, where $i = 1,2,3$. All possible schedules are described by the behavior of the plant $G_1 \| G_2 \| G_3$, where G_1, G_2, G_3 are nonblocking generators with $L_m(G_i) = \{(r_i a_i e_i)^j \mid j \in \mathbb{Z}_+\}$, which is also denoted as $(r_i a_i e_i)^*$, and the set of controllable events is $\Sigma_c = \{a_i \mid i = 1,2,3\}$. The specification K (Fig. 49.1) describes the correct behavior consisting in finishing the transaction in the exit stage before another transaction can proceed to the exit phase.

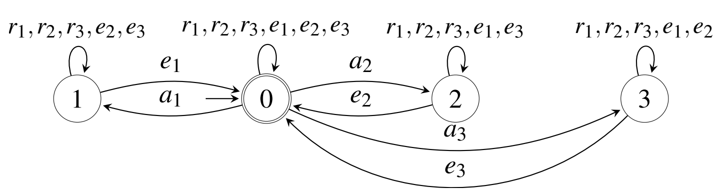

Fig. 49.1 Specification K

49.3 Concepts

The reader is referred to Chapter **??** for the basic notions and concepts of discrete-event systems and supervisory control. Having a global specification, the first step we need to do is to identify the right parts of the specification corresponding to each of the respective subsystems.

A language K is *conditionally decomposable* with respect to event sets Σ_1, Σ_2, Σ_k, where $\Sigma_1 \cap \Sigma_2 \subseteq \Sigma_k \subseteq \Sigma_1 \cup \Sigma_2$, if

$$K = P_{1+k}(K) \parallel P_{2+k}(K)\,,$$

where $P_{i+k} : (\Sigma_1 \cup \Sigma_2)^* \to (\Sigma_i \cup \Sigma_k)^*$ is a projection, for $i = 1,2$.

There always exists an extension of Σ_k that satisfies this condition; $\Sigma_k = \Sigma_1 \cup \Sigma_2$ is such a trivial example. A polynomial algorithm to check whether the condition is satisfied and, if not, to extend the event set Σ_k so that it becomes satisfied can be found in [9]. The question which extension is the most appropriate requires further investigation. To find the minimal extension with respect to set inclusion is an NP-hard problem [8].

For event sets $\Sigma_i, \Sigma_j, \Sigma_\ell \subseteq \Sigma$, in what follows we use the notation P_ℓ^{i+j} to denote the projection from $(\Sigma_i \cup \Sigma_j)^*$ to Σ_ℓ^*. If $\Sigma_i \cup \Sigma_j = \Sigma$, we simplify the notation to P_ℓ. Moreover, $\Sigma_{i,u} = \Sigma_i \cap \Sigma_u$ denotes the set of locally uncontrollable events of the event set Σ_i.

Languages K and L are *synchronously nonconflicting* if $\overline{K \parallel L} = \overline{K} \parallel \overline{L}$.

Lemma 49.1 ([8]). *Let K be a language. If the language $\overline{K}$ is conditionally decomposable, then the languages $P_{1+k}(K)$ and $P_{2+k}(K)$ are synchronously nonconflicting.*

49.4 Construction of Coordinator

In the statement of the problem above, we have mentioned the notion of a coordinator. The fundamental problem, however, is the construction of such a coordinator. We now discuss one of the possible constructions of a suitable coordinator.

Algorithm 1 (Construction of a coordinator) *Consider two subsystems G_1 and G_2 over the event sets Σ_1 and Σ_2, respectively, and let K be a specification language. Construct an event set Σ_k and a coordinator G_k as follows:*

1. *Set $\Sigma_k = \Sigma_1 \cap \Sigma_2$ to be the set of all shared events.*
2. *Extend Σ_k with events of $\Sigma_1 \cup \Sigma_2$ so that K and $\overline{K}$ are conditionally decomposable (for instance using a method described in [9]).*
3. *Set the coordinator $G_k = P_k(G_1) \parallel P_k(G_2)$; for a generator G and a projection P, $P(G)$ is a generator whose behavior satisfies $L(P(G)) = P(L(G))$ and $L_m(P(G)) = P(L_m(G))$.*

Example 49.2. Consider the statement of Example 49.1. We can verify that for $\Sigma_k = \{a_1, a_2, a_3\}$, the specification language K and its prefix closure $\overline{K}$ are conditionally decomposable with respect to $\Sigma_1, \Sigma_2, \Sigma_3$ and Σ_k. The coordinator is then computed as $G_k = P_k(G_1) \| P_k(G_2) \| P_k(G_3)$.

From the complexity viewpoint, the problem is that the projected generator $P_k(G_i)$ can have exponential number of states compared to the generator G_i. So far, the only known condition ensuring that the projected generator is smaller (in the number of states) than the original one is the observer property (see Definition 49.1 below). Therefore, we might need to add step (2b) to further extend Σ_k so that the projection P_k is an $L(G_i)$-observer, for $i = 1, 2$. A polynomial algorithm how to do this can be found in [16, 2].

Definition 49.1 (Observer property). Let $\Sigma_k \subseteq \Sigma$. The projection $P_k : \Sigma^* \to \Sigma_k^*$ is an *L-observer* for a language $L \subseteq \Sigma^*$ if for every $t \in P(L)$ and $s \in \overline{L}$, if $P(s)$ is a prefix of t, then there exists $u \in \Sigma^*$ such that $su \in L$ and $P(su) = t$, cf. Fig. 49.2.

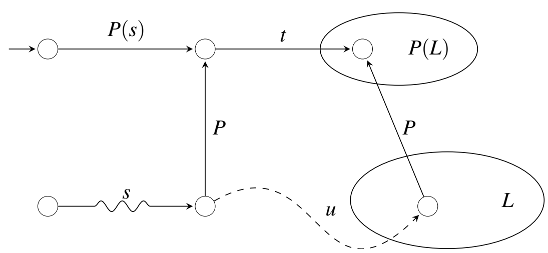

Fig. 49.2 Demonstration of the observer property

Example 49.3. The projection P_k from Example 49.2 is a K-observer, but it is not an $L_m(G_i)$-observer for $i = 1, 2, 3$. However, the projected generators $P_k(G_i)$, $i = 1, 2, 3$, have only one state.

Theorem 49.1. *If a projection P is an $L(G)$-observer, for a generator G, then the minimal generator for the language $P(L(G))$ has no more states than G.*

Based on this result, the coordinator G_k is expected to be quite small compared to the global plant $G_1 \| G_2$.

49.5 Theory

The theory presented here is based on the latest results that can be found in [7] (see also [8]), together with the results from [10].

Let G_1 and G_2 be two generators over Σ_1 and Σ_2, respectively, and let G_k be a coordinator over Σ_k. A language $K \subseteq L(G_1 \| G_2 \| G_k)$ is *conditionally controllable* for generators G_1, G_2, G_k and uncontrollable event sets $\Sigma_{1,u}$, $\Sigma_{2,u}$, $\Sigma_{k,u}$ if

1. $P_k(K)$ is controllable with respect to $L(G_k)$ and $\Sigma_{k,u}$,
2. $P_{1+k}(K)$ is controllable with respect to $L(G_1) \parallel \overline{P_k(K)}$ and $\Sigma_{1+k,u}$,
3. $P_{2+k}(K)$ is controllable with respect to $L(G_2) \parallel \overline{P_k(K)}$ and $\Sigma_{2+k,u}$,

where $\Sigma_{i+k,u} = (\Sigma_i \cup \Sigma_k) \cap \Sigma_u$, for $i = 1,2$.

Example 49.4. Consider Example 49.2. It can be verified that $P_k(K) = \{a_1, a_2, a_3\}^*$ is controllable with respect to $L(G_k) = P_k(K)$ and $\Sigma_{k,u} = \emptyset$. This does not hold for $P_{i+k}(K)$ because the language is not included in $L(G_i) \| \overline{P_k(K)}$, for $i = 1,2,3$.

As in the monolithic case, we need a notion similar to $L_m(G)$-closedness. A nonempty language $K \subseteq \Sigma^*$ is *conditionally closed* for generators G_1, G_2, G_k if

1. $P_k(K)$ is $L_m(G_k)$-closed,
2. $P_{1+k}(K)$ is $L_m(G_1) \parallel P_k(K)$-closed,
3. $P_{2+k}(K)$ is $L_m(G_2) \parallel P_k(K)$-closed.

Example 49.5. Consider Example 49.2. It can be verified that $P_k(K)$ is $L_m(G_k)$-closed, but $P_{i+k}(K)$ is not $L_m(G_i) \| P_k(K)$-closed, for $i = 1,2,3$.

If the specification K is conditionally closed and conditionally controllable, then there exists a nonblocking supervisor S_k such that $L_m(S_k/G_k) = P_k(K)$, which follows from the basic theorem of supervisory control applied to languages $P_k(K)$ and $L(G_k)$, see [1] or Chapter **??**.

Theorem 49.2. *Consider the problem specified above. There exist nonblocking supervisors S_1, S_2, S_k solving the problem if and only if the specification language K is both conditionally controllable with respect to G_1, G_2, G_k and $\Sigma_{1,u}$, $\Sigma_{2,u}$, $\Sigma_{k,u}$, and conditionally closed with respect to G_1, G_2, G_k.*

Example 49.6. Consider Example 49.2. According to Examples 49.4 and 49.5, there do not exist such supervisors that would reach the specification K.

If the specification is not conditionally controllable, we can compute the supremal conditionally-controllable sublanguage.

Theorem 49.3. *The supremal conditionally controllable sublanguage of a specification language always exists and is equal to the union of all conditionally controllable sublanguages of the specification.*

Consider the problem specified above and define the languages

$$\begin{aligned} \sup C_k &= \sup C(P_k(K), L(G_k), \Sigma_{k,u}), \\ \sup C_{1+k} &= \sup C(P_{1+k}(K), L(G_1) \| \overline{\sup C_k}, \Sigma_{1+k,u}), \\ \sup C_{2+k} &= \sup C(P_{2+k}(K), L(G_2) \| \overline{\sup C_k}, \Sigma_{2+k,u}). \end{aligned} \tag{49.1}$$

Example 49.7. Consider Example 49.2. We can compute $\sup C_k$ (Fig. 49.4) and $\sup C_{1+k}$, $\sup C_{2+k}$, $\sup C_{3+k}$ depicted in Fig. 49.3.

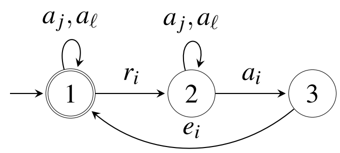

Fig. 49.3 Supervisor $\sup C_{i+k}$, $\{i,j,\ell\} = \{1,2,3\}$.

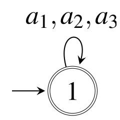

Fig. 49.4 Coordinator.

For the languages defined in (49.1), it always holds that $P_k(\sup C_{i+k}) \subseteq \sup C_k$, for $i = 1,2$. If the converse inclusion also holds, we obtain the supremal conditionally-controllable sublanguage.

Theorem 49.4. *Consider the languages defined in (49.1). If* $\sup C_k \subseteq P_k(\sup C_{i+k})$*, for* $i = 1,2$*, then the language* $\sup C_{1+k} \| \sup C_{2+k}$ *is the supremal conditionally-controllable sublanguage of* K*.*

Example 49.8. Consider the coordinator and supervisors computed in Example 49.7. We can verify that the assumptions of Theorem 49.4 are satisfied. As the language $\sup C_k$ is $L_m(G_k)$-closed and $\sup C_{i+k}$ is $L_m(G_i) \| \sup C_k$-closed, for $i = 1,2,3$, they form a solution for the database problem by Theorems 49.4 and 49.2.

49.6 Coordinator for Nonblockingness

In this section we discuss the coordinator for nonblockingness in the coordination control framework. Recall first that a generator G is nonblocking if $\overline{L_m(G)} = L(G)$.

Theorem 49.5. *Consider languages* L_1 *over* Σ_1 *and* L_2 *over* Σ_2*, and let the projection* $P_0 : (\Sigma_1 \cup \Sigma_2)^* \to \Sigma_0^*$*, with* $\Sigma_1 \cap \Sigma_2 \subseteq \Sigma_0$*, be an* L_i*-observer, for* $i = 1,2$*. Let* G_0 *be a nonblocking generator with* $L_m(G_0) = P_0(L_1) \parallel P_0(L_2)$*. Then the language* $L_1 \| L_2 \| L_m(G_0)$ *is nonblocking, that is,* $\overline{L_1 \| L_2 \| L_m(G_0)} = \overline{L_1} \| \overline{L_2} \| \overline{L_m(G_0)}$*.*

This result is used in the coordination control synthesis as follows. Local supervisors $\sup C_{1+k}$ and $\sup C_{2+k}$ are computed as in (49.1) and the properties of Theorem 49.4 are verified. If they are satisfied, the computed supervisors are the solution of the problem. However, they can still be blocking. In such a case, we can choose the language $L_C = P_0(\sup C_{1+k}) \parallel P_0(\sup C_{2+k})$, where the projection P_0 is a $\sup C_{i+k}$-observer, for $i = 1,2$, (actually, we take the supremal controllable sublanguage of $P_0(\sup C_{1+k}) \parallel P_0(\sup C_{2+k})$, cf. [8]) and obtain that the equality

$$\begin{aligned} \overline{\sup C_{1+k} \parallel \sup C_{2+k} \parallel L_C} &= \overline{\sup C_{1+k} \parallel \sup C_{2+k}} \\ &= \overline{\sup C_{1+k}} \parallel \overline{\sup C_{2+k}} \parallel \overline{L_C} \end{aligned}$$

holds by Theorem 49.5. In other words, L_C is the behavior of a nonblocking coordinator. This gives the following algorithm.

Algorithm 2 (Coordinator for nonblockingness) *Consider the notation above.*

1. *Compute* $\sup C_{1+k}$ *and* $\sup C_{2+k}$ *as defined in (49.1).*
2. *Let* $\Sigma_0 := \Sigma_k$ *and* $P_0 := P_k$.
3. *Extend* Σ_0 *so that the projection* P_0 *is both a* $\sup C_{1+k}$*- and a* $\sup C_{2+k}$*-observer.*
4. *Define the coordinator* C *as the minimal nonblocking generator such that* $L_m(C) = \sup C(P_0(\sup C_{1+k}) \parallel P_0(\sup C_{2+k}),\ \overline{P_0(\sup C_{1+k})} \parallel \overline{P_0(\sup C_{2+k})},\ \Sigma_{0,u})$.

Example 49.9. Consider the solution of the database problem computed in Example 49.7. It can be verified that the language $\sup C_{1+k} \| \sup C_{2+k} \| \sup C_{3+k}$ is nonblocking, hence we do not need a coordinator for nonblockingness in this example.

49.7 Prefix-Closed Languages

Here we assume that the specification is prefix-closed. The following notion is required. More details, an explanation and examples can be found in [16].

Definition 49.2 (Local control consistency). Let L be a prefix-closed language over Σ, and let $\Sigma_0 \subseteq \Sigma$. The projection $P_0 : \Sigma^* \to \Sigma_0^*$ is *locally control consistent* (LCC) with respect to $s \in L$ if for all $\sigma_u \in \Sigma_0 \cap \Sigma_u$ such that $P_0(s)\sigma_u \in P_0(L)$, it holds that either there does not exist any $u \in (\Sigma \setminus \Sigma_0)^*$ such that $su\sigma_u \in L$, or there exists $u \in (\Sigma_u \setminus \Sigma_0)^*$ such that $su\sigma_u \in L$. The projection P_0 is LCC with respect to a language L if P_0 is LCC for all words of L.

Consider generators G_1, G_2, G_k, and denote $L_i = L(G_i)$, for $i = 1,2,k$. There is not yet a general procedure to compute the supremal conditional controllable sublanguage. However, there is a procedure for prefix-closed specifications.

Theorem 49.6. *Let* $K \subseteq L_1 \| L_2 \| L_k$ *be a prefix-closed language over* $\Sigma_1 \cup \Sigma_2 \cup \Sigma_k$*, where* $L_i = \overline{L_i} \subseteq \Sigma_i^*$*,* $i = 1,2,k$*. Assume that the language* K *is conditionally decomposable and consider the languages defined in (49.1). Let the projection* P_k^{i+k} *be an* $(P_i^{i+k})^{-1}(L_i)$*-observer and LCC for* $(P_i^{i+k})^{-1}(L_i)$*, for* $i = 1,2$*. Then* $\sup C_{1+k} \| \sup C_{2+k}$ *is the supremal conditionally-controllable sublanguage of* K*.*

The following corollary explains the relation to the notion of controllability of the monolithic case.

Corollary 49.1. *In the setting of Theorem 49.6, the supremal conditionally-controllable sublanguage of* K *is controllable with respect to* $L_1 \| L_2 \| L_k$ *and* Σ_u*.*

Finally, the last theorem states the conditions under which the solution is optimal.

Theorem 49.7. *Consider the setting of Theorem 49.6. If, in addition,* $L_k \subseteq P_k(L)$ *and* P_{i+k} *is LCC for* $P_{i+k}^{-1}(L_i \| L_k)$*, for* $i = 1,2$*, then* $\sup C(K, L_1 \| L_2 \| L_k, \Sigma_u)$ *is the supremal conditionally-controllable sublanguage of* K*.*

49.8 Further Reading

The theory presented here is based on paper [7]. This topic is still under investigation. For other structural conditions on local plants under which it is possible to synthesize the supervisors locally, but which are quite restrictive, see [3, 12]. Among the most successful approaches to supervisory control of distributed discrete-event systems are those that combine distributed and hierarchical control [16, 17], or the approach based on interfaces [13]. For coordination control of linear or stochastic systems, the reader is referred to [4, 15].

References

1. C. G. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Springer, 2008.
2. L. Feng and W.M. Wonham. On the computation of natural observers in discrete-event systems. *Discrete Event Dyn. Syst.*, 20:63–102, 2010.
3. B. Gaudin and H Marchand. Supervisory control of product and hierarchical discrete event systems. *Eur. J. Control*, 10(2):131–145, 2004.
4. P. L. Kempker, A. C. M. Ran, and J. H. van Schuppen. Construction of a coordinator for coordinated linear systems. In *ECC 2009*, pages 4979–4984, 2009.
5. J. Komenda, T. Masopust, and J. H. van Schuppen. Coordinated control of discrete event systems with nonprefix-closed languages. In *IFAC World Congress*, pages 6982–6987, 2011.
6. J. Komenda, T. Masopust, and J. H. van Schuppen. Synthesis of controllable and normal sublanguages for discrete-event systems using a coordinator. *Systems Control Lett.*, 60(7):492–502, 2011.
7. J. Komenda, T. Masopust, and J. H. van Schuppen. On algorithms and extensions of coordination control of discrete-event systems. In *WODES*, pages 245–250, 2012.
8. J. Komenda, T. Masopust, and J. H. van Schuppen. Coordination Control of Discrete-Event Systems Revisited. Extended version of [7]. Submitted for publication, 2013.
9. J. Komenda, T. Masopust, and J. H. van Schuppen. On conditional decomposability. *Systems Control Lett.*, 61(12):1260–1268, 2012.
10. J. Komenda, T. Masopust, and J. H. van Schuppen. Supervisory control synthesis of discrete-event systems using a coordination scheme. *Automatica*, 48(2):247–254, 2012.
11. J. Komenda and J. H. van Schuppen. Coordination control of discrete event systems. In *WODES*, pages 9–15, 2008.
12. J. Komenda, J. H. van Schuppen, B. Gaudin, and H. Marchand. Supervisory control of modular systems with global specification languages. *Automatica*, 44(4):1127–1134, 2008.
13. R. J. Leduc, D. Pengcheng, and S. Raoguang. Synthesis method for hierarchical interface-based supervisory control. *IEEE Trans. Automat. Control*, 54(7):1548–1560, 2009.
14. Th. Moor et al. libFAUDES – a discrete event systems library, 2012. [Online]. Available at http://www.rt.eei.uni-erlangen.de/FGdes/faudes/.
15. A. C. M. Ran and J. H. van Schuppen. Control for coordination of linear systems. In *MTNS 2008*, Blacksburg, USA, 2008.
16. K. Schmidt and C. Breindl. Maximally permissive hierarchical control of decentralized discrete event systems. *IEEE Trans. Automat. Control*, 56(4):723–737, 2011.
17. K. Schmidt, Th. Moor, and S. Perk. Nonblocking hierarchical control of decentralized discrete event systems. *IEEE Trans. Automat. Control*, 53(10):2252–2265, 2008.
18. W. M. Wonham. Supervisory control of discrete-event systems. Lecture notes, Department of electrical and computer engineering, University of Toronto, 2009.