PERMISSIVITE MAXIMALE DU CONTROLE DES SYSTEMES A EVENEMENTS DISCRETS

30/09/2017
Auteurs : Stéphane Riedweg, Sophie Pinchinat
Publication e-STA N° e-STA 2004-3
DOI :

Télécharger Le téléchargement implique l'acceptation de nos conditions d'utilisation

e-STA 2004-3

• Abstract
• Authors
• Metrics
• License

Abstract

Authors

Stéphane Riedweg

PERMISSIVITE MAXIMALE DU CONTROLE DES SYSTEMES A EVENEMENTS DISCRETS

Sophie Pinchinat

PERMISSIVITE MAXIMALE DU CONTROLE DES SYSTEMES A EVENEMENTS DISCRETS

Metrics

Downloaded: 15
Viewed: 4
Size: 108.56 KB
Type: application/pdf
La SEE (Société de l'Electricité, de l'Electronique et des Technologies de l'Information et de la Communication – Association reconnue d'utilité publique, régie par la loi du 1er juillet 1901) met à la disposition de ses adhérents et des abonnés à ses publications, un ensemble de documents numériques accessibles à partir de son portail des publications. Ces documents incluent notamment les articles des revues REE, 3 EI et e-STA disponibles sous forme numérique ainsi que des publications additionnelles regroupées dans l’espace eREE. Les présentes conditions précisent les conditions de diffusion et d’utilisation de ces documents et des informations qu’ils contiennent. L’accès à ces documents, qu’il se fasse de façon gratuite ou dans le cadre d’abonnements ou d’achats faits à titre onéreux, implique l’acceptation sans restriction de ces dispositions.

**Droits de propriété et de diffusion des contenus téléchargés sur le portail des publications**

Les contenus rendus accessibles sur le portail des publications sont, en règle générale, protégés par le droit d’auteur. En tant que producteur, et le cas échant d’auteur, des informations rassemblées dans les contenus accessibles par ce portail, SEE se réserve l’exclusivité des droits de copie et de diffusion de tout ou partie de ces contenus.

Les contenus sont rendus accessibles à titre individuel, pour les besoins de la personne en détenant des droits d’accès en cours de validité. Aussi, la modification, la reproduction et/ou la diffusion via Internet ou le Web, intranet, extranet ou toute autre forme numérique ou imprimée, de tout ou partie des contenus téléchargés sont interdites. Une tolérance est consentie quant à la reproduction d’extraits limités de ces contenus, dans le cadre de travaux ou d’activités auxquels ils sont utiles, à la condition que l’origine de ces reproductions partielles soit mentionnée de façon lisible et sans ambiguïté. Figureront en particulier : la REE (ou toute autre revue accessible sur le portail) en tant que la source, la référence de la publication et le nom de l’auteur (s’il figure dans la revue).

Ces dispositions s’appliquent également aux figures, illustrations, logos ou images.

**Publication externe des contenus du portail des publications**

Tout extrait des contenus du portail destiné à être utilisé dans des publicités, des communiqués de presse ou du matériel de promotion nécessite un accord préalable écrit de la SEE. Une version préliminaire du document proposé contenant ces extraits doit accompagner chacune de ces demandes. SEE se réserve le droit de refuser un tel usage externe pour quelque raison que ce soit.

**Responsabilités**

La SEE apporte tout le soin possible à la préparation des informations délivrées dans les contenus produits. Cependant elle ne peut être tenue pour responsable d’aucune perte ou frais qui pourrait résulter d’imprécisions, d’inexactitudes, d’erreurs ou de possibles omissions portant sur des
informations publiées, ni des résultats obtenus par l'utilisation et la pratique des informations délivrées.

Utilisation des informations recueillies lors du téléchargement de contenu

Le portail des publications est susceptible d'utiliser des « cookies » afin notamment de permettre l'utilisation de paniers d'achat et de personnaliser les parcours sur le site. SEE se réserve la possibilité d'utiliser les informations recueillies lors des téléchargements pour ses besoins internes et notamment pour l'amélioration de ses services, sans qu'elles puissent être cédées à des partenaires commerciaux. Conformément à la loi "informatique et libertés" du 6 janvier 1978, chaque utilisateur du portail dispose d'un droit d'accès et de rectification aux informations qui le concernent. Pour exercer ce droit, les utilisateurs doivent s'adresser à SEE – 17 rue de l’amiral Hamelin – 75783 Paris Cedex 16, par simple lettre ou en utilisant le formulaire de contact disponible sur son site.

Paris, le 28 avril 2013

Metadata Datacite XML candidate:

```
<resource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://datacite.org/schema/kernel-4"
    <identifier identifierType="DOI">10.23723/545:2004-3/20051</identifier>
    <creators>
        <creator><creatorName>Stéphane Riedweg</creatorName></creator>
        <creator><creatorName>Sophie Pinchinat</creatorName></creator>
    </creators>
    <titles>
        <title>PERMISSIVITE MAXIMALE DU CONTROLE DES SYSTEMES A EVENEMENTS DISCRETS</title>
    </titles>
    <publisher>SEE</publisher>
    <publicationYear>2017</publicationYear>
    <resourceType resourceTypeGeneral="Text">Text</resourceType>
    <dates>
        <date dateType="Created">Sat 30 Sep 2017</date>
        <date dateType="Updated">Sat 30 Sep 2017</date>
        <date dateType="Submitted">Sat 2 Nov 2019</date>
    </dates>
    <alternateIdentifiers>
        <alternateIdentifier alternateIdentifierType="bitstream">340792563bfd4c1a96094089217b7916b622b9e3</alternateIdentifier>
    </alternateIdentifiers>
    <formats>
        <format>application/pdf</format>
    </formats>
    <version>34068</version>
    <descriptions>
        <description descriptionType="Abstract"/>
    </descriptions>
</resource>
```

PERMISSIVITE MAXIMALE DU CONTROLE DES SYSTEMES A EVENEMENTS DISCRETS (MAXIMALLY PERMISSIVE CONTROLLERS IN ALL CONTEXTS) Stéphane Riedweg, Sophie Pinchinat IRISA-INRIA, F-35042, Rennes, France {sriedweg,pinchina}@irisa.fr Abstract: Nous proposons un formalisme logique pour la spécification des problèmes de contrôle dans laquelle la requête de permissivité maximale des contrôleurs est rendue explicite. L'approche utilise une logique modale avec des quantifications aux propositions ; ce cadre logique offre de plus les algorithmes pour la décision des spécifications et, le cas échéant, pour la synthèse des contrôleurs solutions. We propose a logical formalism for the supervisory control specifications where the maximal permissiveness is explicitly handled, independently of the plant. The approach relies on a modal logic with quantified propositions and additionally provides a powerful machinery for decision and synthesis procedures. Keywords: Supervision, Discrete-event Systems, Permissiveness, Optimal Control, Temporal Logic 1. INTRODUCTION The control theory for discrete event systems was initiated by (Ramadge and Wonham, 1989), followed by (Thistle and Wonham, 1994) and many others. More recently, temporal logics specifications were considered by (Kupferman et al., 2000) and (Arnold et al., 2003) in order to solve a larger class of problems. Typically, control problems require to find a supervisory control of the plant such that its (operational) behavior meets some
desired properties, often called the control objectives. In regard to the ob- jectives, they most often concern standard proper- ties such as admissibility, non- blocking, safety etc. Among those, the fact that the controllers should allow any sound event, hence fulfilling the maxi- mal permissiveness property, is generally taken for granted. As studied in depth by (Ramadge and Wonham, 1989) and others, the theory of regular languages offers clear results on the subject: the existence of a controller implies the existence of a unique maximally permissive one. The proposed algorithms hence synthesize precisely this optimal solution. Unfortunately, this cozy situation does not extend to larger frameworks: for example, the ω-closure assumption is necessary for ω-regular definable objectives (as in (Thistle and Wonham, 1994)), the set of control patterns of (Golaszewski and Ramadge, 1987) must be union-closed, etc. Worse is the case of branching-time logic speci- fications as in (Kupferman et al., 2000; Arnold et al., 2003) where neither the unicity and nor the existence of a maximally permissible controller are guaranteed anymore. For instance, there is no optimal supervision of the plant (a + b)ω which eventually disables event b.

Surprisingly, to our knowledge, the maximal permissiveness property has never been approached on its own, like a pos- sible parameter of the supervisory control prob- lems, likely because regular language semantics withstand this subject. In this paper, we propose a formalism which allows an explicit handling of maximally permissiveness, in addition to standard supervi- sory control specifications. The approach relies on a logical framework and provides a powerful machinery for decision processes and effective synthesis. The logic is an extension of the propositional mu-calculus (Kozen, 1983; Arnold and Niwinski, 2001), called the Quantified mu-calculus (qLµ), as originally proposed by (Riedweg and Pinchiniat, 2003b). The extension feature relies on the use of quantifications over (atomic) propositions. The resulting logic remains decidable as well as its model-checking thanks to its equivalence with infi- nite tree- automata, as already the case for the mu- calculus (see (Emerson and Lei, 1986)). For this reason, effective procedures to synthesis supervi- sory controllers are made possible by computing a finite model of a (infinite) tree-automaton (see also (Riedweg, 2003)). In this paper, we do not aim at explaining the synthesis procedure, but instead we shall take it for granted from (Riedweg and Pinchiniat, 2003b). Rather, we demonstrate how the logical formalism elegantly specifies supervisory control problems with an explicit mention of the maximal per- missiveness property inside. Moreover, we make a proof that no other framework known in the literature can reach this expressive power. The paper is organized as follows: Section 2 introduces the logic qLµ. Section 3 explains how control objectives can be specified. Next, Section 4 dedicates to the maximal permissiveness specifi- cation with a proof of correctness. In Section 5, expressiveness issues are discussed, followed by a section of concluding remarks. 2. QUANTIFIED MU-CALCULUS Models are deterministic finite state machines with marked states, called processes in Defini- tion 2, as plants in supervisory control problems normally are. Since branching-time temporal sta- tements will be considered, the natural notion of behavior is the execution tree, namely the (possibly infinite) unfolding of the finite state machine. Given that controllers aim at pruning the execu- tion tree, the proposed logic can state in particular the existence of some pruning of the tree that sat- isfies a (mu- calculus definable) desired property. The prunings are simply represented by placing a fresh atomic proposition on the tree to delimit the remaining subtree after pruning: in this way, an edge of the tree where the source node is positively labeled by the proposition whereas the target node is not is meant to be pruned. Technically, an atomic proposition p is added to the unfolding of a process by composing the pro- cess with a particular one, namely a p-labeling process E as in Definition 3; controllers are ac- tually derived from those. To get the tree pruning induced from the proposition done, we prune the labeling process according to Definition 5 to get E(p), before we compose it with the system. So that, as stated by Proposition 7, wondering whether a given pruning/control S × E(p) of the system verifies some desired property α reduces to wonder whether the corresponding p-labeling by S × E is appropriated satisfies some adjustment α ∗ p of the original property. We assume given a finite set of events Σ = {a, b, . . . }, a set of atomic propositions AP = {p, p0, c, c0, . . . }, and a set of variables Var = {X, Y, . . . }. Definition 1. Syntax of qLµ We first recall the syntax of the pure mu-calculus, written Lµ. The set of formulas of Lµ is defined by the following grammar: > | p | X | ¬ | ∩ | ∪ | µX(X) where a ∈ Σ, p ∈ AP and X ∈ Var. Fix-points formulas µX(X) can properly be in- terpreted (in Definition 4) whenever each occur- rence of X in µX(X) is under an even number of negation symbols ¬. The quantified mu-calculus, written qLµ, extends Lµ as follows. The set of formulas of qLµ is defined by: Ψ,α | ¬ | α ∨ α0 | µX where p ∈ AP and β ∈ Lµ. Freely extending the classical terminology of the mu-calculus to the quantified mu-calculus, we name sentences all the formulas which each oc- currence of a variable X is binded by a fix-point symbol μ. Also, we write ↓, [α]a, α∧ a, ν, ν(X), and ∀p.α respectively for →, ¬, ¬∧ α, ν∧ α0, ¬ νX ∧ ν¬(X) and ¬∃p.¬ α, as well as a → , [α]a and a ∧ a0 respectively for >, ν ∈Σ[α]a and μ ∈ νX. Lastly, for β ∈ Lµ, INV (β) is a notation for νX.[X ∧ μX: according to Definition 4 further, it states “from now on, the property β always holds”. Since in general, fixed-point operators and quanti- fier do not commute, we forbid the use of quantifi- cations inside fixed-point terms, all the more since the formulas will have enough expressive power for control problems specifications, as shown in the next section. The semantics of the quantified mu-calculus (Def- inition 4) relies on models called processes: Definition 2. (Processes) Given a finite set Γ ⊆ AP, a process on Γ is a tuple S = s0, t, Li, where S is a set of states, s0 is the initial state, t : S × S → S is a partial function called the transition function and L : S → 2Γ labels states by propositions. A process S is finite if S is finite and it is complete if t(s, a) is defined everywhere. Processes are normally subject to a natural abs- traction according to their execution tree isomor- phism class. Two processes with isomorphic exe- cution trees are called bisimilar. As announced, processes are composed in a syn-chronous manner, as to diminish their behavior: The (synchronous) product of S1 = s1, t0, t1, L1 on Γ1 and S2 = s2, t0, t2, L2 on Γ2 (with disjoint Γ1 and Γ2) is S1 × S2 = s1s2, t0, L1 ∪ L2 where: (1) t(s1s2, s2) = (s1t0, s2) whenever s1 = t(t1(s1, a) and s2 = t2(s2, a)), and (2) L1(s1, s2) = L1(s1) ∪ L2(s2). Definition 3. (Labeling Processes) Given p ∈ AP, a p-labeling process is a complete process on [p]. We let Labp be the set of p-labeling processes, and we use E, E0 for typical elements. A labeling of S by p, or a p-labeling of S, is a product S × E, where E ∈ Labp. The logic can now be interpreted. Given a process S, a formula α denotes a subset of the states, those which “satisfy” it. The semantics is given by induction over a, hence the need
to interpret variable formulas: the standard manner consists in setting a valuation val: V ar → P(S) which defines the subset of states denoted by the formulas X ∈ V ar. The pure mu-calculus formulas semantics is standard. Definition 4. (Semantics of $q\mu L$) Given a pro- cess $S = (S, s_0, T, L)$ and a valuation $val: V ar \rightarrow \{0, 1\}$, the $\mu$-calculus formula $\varphi$ is interpreted on $S$ by $\varphi$ = $\mu X. \varphi_2$. As shown by Proposition 7, objects of type $E(\varphi_2)$ is a preorder. Note that if two processes $S_1$ and $S_2$ are bisimilar, then $S_1$ and $S_2$ mean $S_1$ and $S_2$ on the one hand and controllable events, by complementary, on the other hand.

Definition 5. (Pruning) Given any process $S = (S, s_0, T, L)$ and a valuation $val: V ar \rightarrow \{0, 1\}$, the pruning of $S$ is $S(p) = (S, s_0, T, L \setminus p)$, where: (1) for all $s \in S$ and $a \in \Gamma$, $t(s, a) = t(s, a)$, (2) $S$ and $S(p)$ are isomorphic, the $\varphi$-adjustments of $S$ to $S(p)$ is bisimilar to $S \times E(\varphi_2)$ and $S \times E(\varphi)$ is bisimilar to $S(p)$. The soundness of the coming logical specifications can be argued by using two kinds of binary relations between processes. The first being equivalence relations: given a process $S$, two $\varphi$-adjustments of $S$ to $S(p)$ by check the simulation $\{(s, s) \in S \times E(\varphi_2) \mid \varphi_2 = \mu X. \varphi_1\}$. Proposition 12. For any $E(S) \times E(\varphi_2)$ is a process $C$ with the following: (1) at any time, it allows all uncontrollable events: $C \models ADm \equiv INV(\neg(c \wedge a))$, and (3) the successors of reachable states via uncontrollable events remain reachable. Proposition 12. For any $E := E(S) \times E(\varphi_2)$, whenever $S \times E(\varphi_2)$ is a process $C$ with the following: (1) at any time, it allows all uncontrollable events: $C \models ADm \equiv INV(\neg(c \wedge a))$, and (3) the successors of reachable states via uncontrollable events remain reachable. Proposition 12.

Definition 5. (Pruning) Given any process $S = (S, s_0, T, L)$ and a valuation $val: V ar \rightarrow \{0, 1\}$, the pruning of $S$ is $S(p) = (S, s_0, T, L \setminus p)$, where: (1) for all $s \in S$ and $a \in \Gamma$, $t(s, a) = t(s, a)$, (2) $S$ and $S(p)$ are isomorphic, the $\varphi$-adjustments of $S$ to $S(p)$ is bisimilar to $S \times E(\varphi_2)$ and $S \times E(\varphi)$ is bisimilar to $S(p)$. The soundness of the coming logical specifications can be argued by using two kinds of binary relations between processes. The first being equivalence relations: given a process $S$, two $\varphi$-adjustments of $S$ to $S(p)$ by check the simulation $\{(s, s) \in S \times E(\varphi_2) \mid \varphi_2 = \mu X. \varphi_1\}$. Proposition 12.

Definition 5. (Pruning) Given any process $S = (S, s_0, T, L)$ and a valuation $val: V ar \rightarrow \{0, 1\}$, the pruning of $S$ is $S(p) = (S, s_0, T, L \setminus p)$, where: (1) for all $s \in S$ and $a \in \Gamma$, $t(s, a) = t(s, a)$, (2) $S$ and $S(p)$ are isomorphic, the $\varphi$-adjustments of $S$ to $S(p)$ is bisimilar to $S \times E(\varphi_2)$ and $S \times E(\varphi)$ is bisimilar to $S(p)$. The soundness of the coming logical specifications can be argued by using two kinds of binary relations between processes. The first being equivalence relations: given a process $S$, two $\varphi$-adjustments of $S$ to $S(p)$ by check the simulation $\{(s, s) \in S \times E(\varphi_2) \mid \varphi_2 = \mu X. \varphi_1\}$. Proposition 12.
by Proposition 7, \( P \times E \) satisfies \( \alpha \rightarrow c \). Assume now that \( P \) satisfies the formula (1). Then there exists \( E \in \text{Labc} \) such that \( P \times E \) satisfies \( \alpha \rightarrow c \) and Controller(c). Since \( P \times E \models \text{Controller}(c) \), there exists (Proposition 12) a labeling process \( E' \) such that \( E' \models \text{Controller}(c) \) and \( \text{Controller}(c) \). Moreover, \( E_0 = P \times E \) and \( P \times E \models \alpha \rightarrow c \) imply \( P \times E \models \alpha \rightarrow c \). Then, take \( E_0 \) to conclude (by Proposition 7). 4. MAXIMALLY PERMISSIVE CONTROLLERS We first give the definition for Maximal Permissiveness in a fully natural way. Next we specify it in \( qL\mu \), with the arguments for its correctness. Definition 14. (Maximally Permissive Controller) Given a formula \( \alpha \in qL\mu \), we say that the controller \( C \) of \( P \) for \( \alpha \) is maximally permissive if for any other controller \( C_0 \) of \( P \) for \( \alpha \), we do not have \( S \times C < S \times C_0 \). Essentially, since the labelings of the plant can be compared regarding the size of the remaining subtree after pruning, so can the underlying controllers, hence our capability to express optimality in the logic: given any two fresh atomic propositions \( c \) and \( \delta \), we define the formulas \( c \rightarrow \delta \) and \( \delta \\rightarrow c \) by: \( c \rightarrow \delta \) and \( \delta \rightarrow c \) contain a \( c \)-adjustment to express that \( c \) holds all along labeled executions. Proposition 15. For all \( E \in \text{Labc} \), we have: \( P \times E(c) \times P \times E(\delta) \) if \( \neg E \times E \models c \rightarrow \delta \). PROOF. Assume that \( P \times E \models c \rightarrow \delta \). By Proposition 7, this is equivalent to say that \( P \times E \models E \times E(c) \) satisfies [ ]\( \text{INV} \) (c \( \delta \)) \( \rightarrow c \rightarrow \delta \). Formula \( c \rightarrow \delta \) contains a \( c \)-adjustment to express that \( c \) holds all along labeled executions. Proposition 15. For all \( E \in \text{Labc} \), we have: \( P \times E(c) \times P \times E(\delta) \) if \( \neg E \times E \models c \rightarrow \delta \). PROOF. Assume that \( P \times E \models c \rightarrow \delta \). By Proposition 7, this is equivalent to say that \( P \times E \models E \times E(c) \) satisfies [ ]\( \text{INV} \) (c \( \delta \)) \( \rightarrow c \rightarrow \delta \). Formula \( c \rightarrow \delta \) contains a \( c \)-adjustment to express that \( c \) holds all along labeled executions. Proposition 15. For all \( E \in \text{Labc} \), we have: \( P \times E(c) \times P \times E(\delta) \) if \( \neg E \times E \models c \rightarrow \delta \). PROOF. Assume that \( P \times E \models c \rightarrow \delta \). By Proposition 7, this is equivalent to say that \( P \times E \models E \times E(c) \) satisfies [ ]\( \text{INV} \) (c \( \delta \)) \( \rightarrow c \rightarrow \delta \). Formula \( c \rightarrow \delta \) contains a \( c \)-adjustment to express that \( c \) holds all along labeled executions. Proposition 15. For all \( E \in \text{Labc} \), we have: \( P \times E(c) \times P \times E(\delta) \) if \( \neg E \times E \models c \rightarrow \delta \). PROOF. Assume that \( P \times E \models c \rightarrow \delta \). By Proposition 7, this is equivalent to say that \( P \times E \models E \times E(c) \) satisfies [ ]\( \text{INV} \) (c \( \delta \)) \( \rightarrow c \rightarrow \delta \). Formula \( c \rightarrow \delta \) contains a \( c \)-adjustment to express that \( c \) holds all along labeled executions. Proposition 15. For all \( E \in \text{Labc} \), we have: \( P \times E(c) \times P \times E(\delta) \) if \( \neg E \times E \models c \rightarrow \delta \). PROOF. Assume that \( P \times E \models c \rightarrow \delta \). By Proposition 7, this is equivalent to say that \( P \times E \models E \times E(c) \) satisfies [ ]\( \text{INV} \) (c \( \delta \)) \( \rightarrow c \rightarrow \delta \). Formula \( c \rightarrow \delta \) contains a \( c \)-adjustment to express that \( c \) holds all along labeled executions. Proposition 15. For all \( E \in \text{Labc} \), we have: \( P \times E(c) \times P \times E(\delta) \) if \( \neg E \times E \models c \rightarrow \delta \). PROOF. Assume that \( P \times E \models c \rightarrow \delta \). By Proposition 7, this is equivalent to say that \( P \times E \models E \times E(c) \) satisfies [ ]\( \text{INV} \) (c \( \delta \)) \( \rightarrow c \rightarrow \delta \). Formula \( c \rightarrow \delta \) contains a \( c \)-adjustment to express that \( c \) holds all along labeled executions. Proposition 15. For all \( E \in \text{Labc} \), we have: \( P \times E(c) \times P \times E(\delta) \) if \( \neg E \times E \models c \rightarrow \delta \). PROOF. Assume that \( P \times E \models c \rightarrow \delta \). By Proposition 7, this is equivalent to say that \( P \times E \models E \times E(c) \) satisfies [ ]\( \text{INV} \) (c \( \delta \)) \( \rightarrow c \rightarrow \delta \). Formula \( c \rightarrow \delta \) contains a \( c \)-adjustment to express that \( c \) holds all along labeled executions.

https://www.see.asso.fr/en/node/20051/landing