Distributed hybrid control synthesis for multi-agent systems from high level specifications

Co^4 Workshop, Toulouse, October 2016

Dimos V. Dimarogonas
(joint work with Jana Tumova, Dimitris Boskos and Meng Guo)

Automatic Control,
KTH Royal Institute of Technology, Sweden

European Research Council
Established by the European Commission

RE CON FIG

WALLENBERG ACADEMY FELLOWS

Vetenskapsrådet

Swedish Foundation for Strategic Research

KTH VETENSKAP OCH KONST

Table of Contents

Background

- Multi-agent control: motivated by a large variety of engineering applications: transportation systems, robotics, smart grids
- Multi-agent control objectives: simple/control type (consensus, formation control, ...)
- Formal methods based planning: higher level objectives for single agent
- Based on discrete represenations (aka abstractions) of control systems

State of the Art

Single Agent-Single Task

$$\dot{X} = f(X, u)$$
$$\varphi = (\Diamond(r_1 \wedge \Diamond r_2)) \wedge (\Diamond\Box r_3)$$

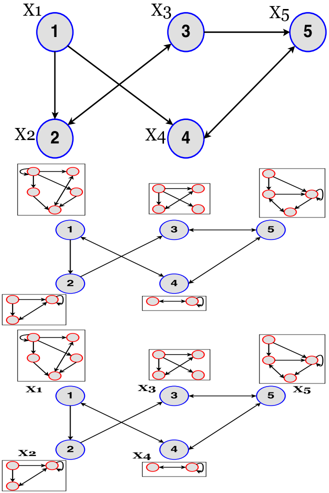
- High-level task specs using formal languages
- Planning on discrete abstraction of agent dynamics
- Implemented by continuous control sequence

Multiple Agents-Multiple Tasks

- Need for distributed, bottom-up solutions to deal with:
    - Distributed tasks and abstractions
    - Couplings, limited communication

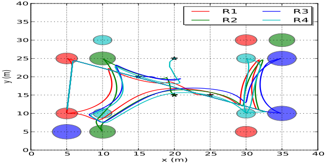
Proposed approach

- Multi-agent control layer: distributed control through continuous state information
- Formal methods based planning: distributed task planning based on discrete information exchange
- Hybrid control: blending continuous and discrete information, need for abstractions of multi-agent control systems



Today's talk

- Task planning and control through specification-based abstraction
- Abstractions of dynamically coupled multi-agent systems
- Distributed task planning for task-level dependencies



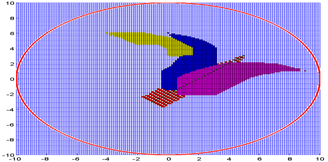


Table of Contents

Problem Formulation

- A team of N mobile agents, $x_i(t), u_i(t) \in \mathbb{R}^2$:

$$\dot{x}_i(t) = u_i(t), \qquad i \in \mathcal{N} = \{1, \cdots, N\}.$$

- Agents i can observe agent j's position $x_j(t)$ only if:

$$\|x_i(t) - x_j(t)\| \leq r.$$

  Initial network G_0.

- Sphere regions of interest: $\mathcal{R}_i = \{R_{i\ell},\, \ell \in \{1, \cdots, M_i\}\}$.
  $R_{i\ell} = (c_{i\ell},\, r_{i\ell})$.

- Assumptions on the workspace.

- Services Σ_i available at each region in $\mathcal{R}_i$.

Problem Formulation, cont'd

- Local LTL task specification φ_i, over Σ_i.
- Note that φ_i can be co-safe or general LTL formulas.
- φ_i specifies the sequences at which the services should be done at certain regions.

Problem

How to synthesize the control input $u_i(t)$ and the discrete plan $\mathbb{S}_i$ such that

$$\varphi_i \text{ is satisfied}, \quad \forall i \in \mathcal{N}$$

and $\|x_i(t) - x_j(t)\| \leq r$, $\forall (i, j) \in E_0$, $\forall t \in [0, \infty)$.

Challenges

- Discrete task planning
- Continuous motion constraints
- Sensing limitations

Solution: three main steps.

- High-level discrete plan synthesis.
- Distributed potential-field-based motion control.
- Hybrid control strategy.

Step1. Discrete Plan Synthesis

Aim

Each agent synthesizes a local discrete plan that satisfies φ_i and minimizes a cost function.

- Automata-based model-checking algorithm[1]
- Discrete plan synthesized locally by each agent $i \in \mathcal{N}$:

$$\mathbb{S}_i = \sigma_{i1} \cdots \sigma_{is_i}(\sigma_{i(s_i+1)} \cdots \sigma_{iN_i})^{\omega}, \qquad \sigma_{is_i} = (R_{is_i}, \Sigma_{is_i}).$$

- Our algorithm minimizes the maximal distance between two consecutive regions along the plan[2].

---

[1] C. Baier, J.-P Katoen. *Principles of model checking*, 2008.

[2] S. L. Smith, J. Tumova, C. Belta, D. Rus. Optimal Path Planning for Surveillance with Temporal Logic Constraints. The International Journal of Robotics Research, 2011.

Step2. Distributed Motion Control

- Setup for motion control:
  - Each agent has its goal region $\sigma_{ig} = (R_{ig}, \Sigma_{ig})$, but only known locally.
  - Relative-distance constraints.

Goal

Design a distributed control law $u_i(t)$ such that one agent arrives at its goal region in finite time, given the relative-distance constraints.

- *Time-varying* connectivity graph $G(t) = (\mathcal{N}, E(t))$, where $E(t) \subseteq \mathcal{N} \times \mathcal{N}$.
  - initially $G(0) = G_0$; dynamically add new edges.

- Solution: the two-mode control law
  (1) the *active* mode:

$$\mathbf{C}_{act}: \quad u_i(t) \triangleq -d_i\, p_i - \sum_{j \in \mathcal{N}_i(t)} h_{ij}\, x_{ij},$$

  (2) the *passive* mode:

$$\mathbf{C}_{pas}: \quad u_i(t) \triangleq -\sum_{j \in \mathcal{N}_i(t)} h_{ij}\, x_{ij},$$

  where $x_{ij} \triangleq x_i - x_j$; $p_i \triangleq x_i - c_{i\mathrm{g}}$; $R_{ig} = (c_{i\mathrm{g}},\, r_{i\mathrm{g}})$.

$$d_i \triangleq \frac{\varepsilon^3}{(\|p_i\|^2 + \varepsilon)^2} + \frac{\varepsilon^2}{2\,(\|p_i\|^2 + \varepsilon)}; \quad h_{ij} \triangleq \frac{r^2}{(r^2 - \|x_{ij}\|^2)^2}$$

- $\varepsilon > 0$ is a key design parameter.
- u_i is local w.r.t. $\mathcal{N}_i(t)$.

Convergence results

Considering a potential-field like Lyapunov function it can be shown that:

- $G(t)$ remains connected.
- There exists a finite time T_f and one active agent $i^\star \in \mathcal{N}_a$, such that $x_j(T_f) \in R_{i^\star \mathrm{g}}$, $\forall j \in \mathcal{N}$.
- All agents will enter $R_{i^\star \mathrm{g}}$, i.e., $x_j \in R_{i^\star \mathrm{g}}$, $\forall j \in \mathcal{N}$.
- The above holds for any number of active agents that $1 \leq N_a \leq N$.

Potential-field-based Design

Consider the following potential-field function:

$$V(x(t)) \triangleq \frac{1}{2}\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}_i(t)}\phi_c(x_{ij}) + b_i\sum_{i\in\mathcal{N}}\phi_g(x_i)$$

- $\phi_c(x_{ij})$ is an attractive potential to agent i's neighbors.
- $\phi_g(\cdot)$ is an attractive force to agent i's goal:
- $b_i = 1$, $\forall i \in \mathcal{N}_a$ and $b_i = 0$, $\forall i \in \mathcal{N}_p$. $\mathcal{N} = \mathcal{N}_a \cup \mathcal{N}_p$.

Connectivity Results

$G(t)$ remains connected.
No existing edges within $E(T_s)$ will be lost.

- Proof shows that $V(t)$ remains bounded for $t \in [T_s, \infty)$.
  New edges might be added but no existing edges will be lost.

Convergence (non-switching case)

Constant sets of passive and active agents. Analysis of the critical points of V:

- Regions around the critical points:

  $$\mathcal{S}_i \triangleq \{\mathbf{x} \in \mathbb{R}^{2N} \mid \|\mathbf{x} - \mathbf{1}_N \otimes c_{i\mathrm{g}}\| \leq r_S(\varepsilon)\}, \qquad \forall i \in \mathcal{N}_a.$$

  Let $\mathcal{S} \triangleq \cup_{i \in \mathcal{N}_a} \mathcal{S}_i$ and $\mathcal{S}^{\neg} \triangleq \mathbb{R}^{2N} \setminus \mathcal{S}$.

- Lemma 1: There exists $\varepsilon_1 > 0$ such that if $\varepsilon < \varepsilon_1$, all critical points of V in $\mathcal{S}^{\neg}$ are non-degenerate saddle points.

- Lemma 2,3: There exists $\varepsilon < \min\{\varepsilon_2, \varepsilon_6\}$ such that regions $\{\mathcal{S}_i\}$ are sufficiently far. Critical points are close to the region center.

Lemma 4: There exists $\varepsilon_{\min} > 0$ such that if $\varepsilon < \varepsilon_{\min}$, all critical points of V within $\mathcal{S}$ are local minima.

Convergence Results

There exists a finite time $T_f \in [T_s, \infty)$ and one active agent $i^\star \in \mathcal{N}_a$, such that $x_j(T_f) \in R_{i^\star \mathrm{g}}$, $\forall j \in \mathcal{N}$, while $\|x_i(t) - x_j(t)\| < r$, $\forall (i, j) \in E(T_s)$ and $\forall t \in [T_s,\ T_f]$.

- The system converge to the set of local minima within $\mathcal{S}_{i^\star}$ for one active agent $i^\star \in \mathcal{N}_a$.
- All agents would enter $R_{i^\star \mathrm{g}}$, i.e., $x_j \in R_{i^\star \mathrm{g}}$, $\forall j \in \mathcal{N}$.
- All edges within $E(T_s)$ will be preserved for all $t > T_s$

The above theorem holds for any number of active agents that $1 \leq N_a \leq N$.

Step3. Hybrid Control: sc-safe LTL task case

Case one

All tasks $\{\varphi_i\}$ are given as sc-safe LTL formulas.

- If φ_i is sc-safe, every agent has a finite plan

$$\tau_i = (R_{i1}, \Sigma_{i1})(R_{i2}, \Sigma_{i2}) \cdots (R_{iN_i}, \Sigma_{iN_i}).$$

Local switching policy

- When R_{ik} is reached, provide the services Σ_{ik} and then set goal to $R_{i(k+1)}$.
- After $(R_{iN_i}, \Sigma_{iN_i})$, set $b_i = 0$ and be passive.

- Guaranteed that $\forall i \in \mathcal{N}$, φ_i is eventually satisfied, and $\|x_i(t) - x_j(t)\| < r$, $\forall (i, j) \in E(0)$ and $\forall t \geq 0$.

Step3. Hybrid Control: general LTL task case

- LTL and mixed sc-safe LTL/LTL tasks can be also tackled under different switching policies
- Account for infiniteness of satisfying plans
- Further ongoing extension to double-integrator dynamics with collision avoidance and quantified specs (MITL and STL formulas)

Four agents with co-safe or general LTL tasks:

Workspace

- $\Pi_1 = \{\pi_{1\mathtt{tl}}, \pi_{1\mathtt{tr}}, \pi_{1\mathtt{br}}, \pi_{1\mathtt{bl}}\}$. $\Sigma_1 = \{\sigma_{11}, \sigma_{12}\}$.
- $\Pi_2 = \{\pi_{2\mathtt{tl}}, \pi_{2\mathtt{tr}}, \pi_{2\mathtt{bl}}\}$. $\Sigma_2 = \{\sigma_{21}, \sigma_{22}, \sigma_{23}\}$.
- $\Pi_3 = \{\pi_{3\mathtt{tr}}, \pi_{3\mathtt{br}}, \pi_{3\mathtt{bl}}\}$. $\Sigma_3 = \{\sigma_{31}, \sigma_{32}, \sigma_{33}\}$.
- $\Pi_4 = \{\pi_{4\mathtt{tl}}, \pi_{4\mathtt{tr}}, \pi_{4\mathtt{br}}, \pi_{4\mathtt{bl}}\}$. $\Sigma_4 = \{\sigma_{41}, \sigma_{42}, \sigma_{43}\}$.

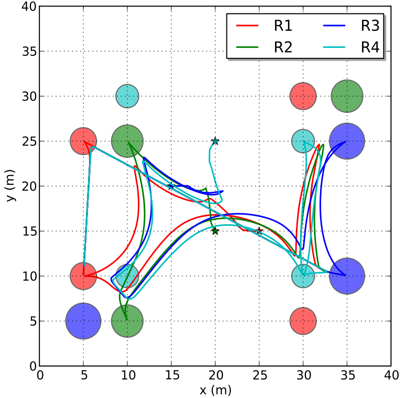
Sc-safe LTL task

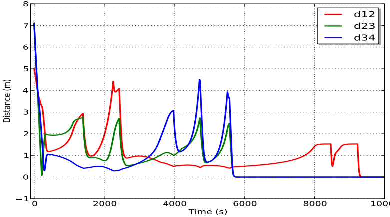
- $\varphi_1 = \Diamond(\sigma_{12} \wedge \Diamond(\sigma_{11} \wedge \Diamond\sigma_{12}))$.
- $\varphi_2 = \Diamond(\sigma_{21} \vee \sigma_{22}) \wedge \Diamond\sigma_{23}$.
- $\varphi_3 = \Diamond(\sigma_{31} \vee \sigma_{32}) \wedge \Diamond\sigma_{33}$.
- $\varphi_4 = \Diamond(\sigma_{42} \wedge \Diamond(\sigma_{41} \wedge \Diamond\sigma_{42}))$.

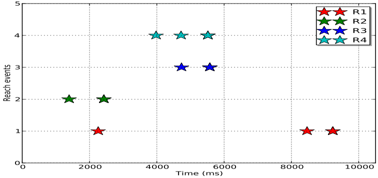
General LTL task

- $\varphi_1 = \Box\Diamond\sigma_{11} \wedge \Box\Diamond\sigma_{12}$.
- $\varphi_2 = \Box\Diamond(\sigma_{21} \vee \sigma_{22} \vee \sigma_{23})$
- $\varphi_3 = \Box\Diamond(\sigma_{31} \vee \sigma_{32} \vee \sigma_{33})$
- $\varphi_4 = \Box\Diamond\sigma_{41} \wedge \Box\Diamond\sigma_{42}$

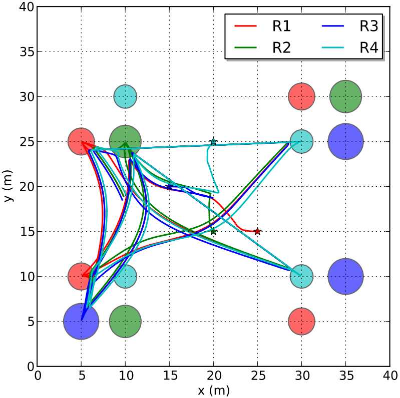
Scenario one

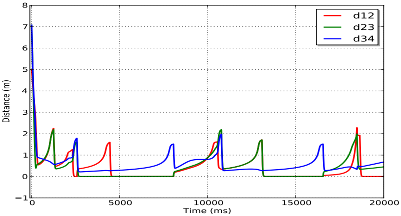






Scenario two





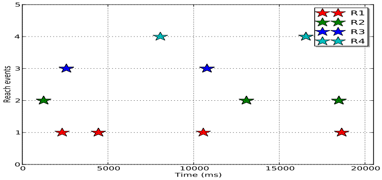


Table of Contents

Motivation

- Coupled multi-agent control systems
- Define discrete representations irrespective of given high-level specs
- May lead to trade-offs or fundamental limits to what can be requested from the system

Systems Description and objective

- Consider the multi-agent system

$$\dot{x}_i = u_i = f_i(x_i, \mathbf{x}_j) + v_i, \mathbf{x}_j = (x_{j_1}, \ldots, x_{j_{N_i}}), i = 1, \ldots, N$$

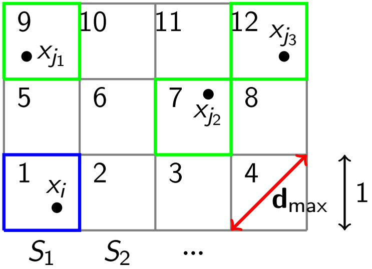
- Closed loop system with coupled constraints $f_i(x_i, \mathbf{x}_j)$ and free inputs v_i
- Goal: abstract continuous space-time system properties in a discrete Transition System
- Goal: find finite abstractions for the multi-agent system in a distributed way that makes sense

Preliminaries - Notation

- Abstraction Requirements: find
  - cell decomposition $\rightarrow$ finite or countable "partition" $\mathcal{S} = \{S_l\}_{l \in \mathcal{I}}$ of the workspace by uniformly bounded sets
  - time step δt
  - which ensure that the discretized model of closed loop system is well posed - meaningful
- Notation
  - Cell Configuration CC of i and its neighbors $j_1, \ldots, j_{N_i}$
  - $N_i + 1$-tuple of cell indices $\mathbf{l}_i = (l_i, l_{j_1}, \ldots, l_{j_{N_i}}) \in \mathcal{I}^{N_i+1}$
- Cell decomposition diameter d_{max}:
  - "maximum" diameter of a cell $S_l \in \mathcal{S}$

$$d_{\max} := \sup\{|x - y| : x, y \in S_l, l \in \mathcal{I}\}$$

Cell Decomposition - Cell Configuration Example



- Cell decomposition: $\mathcal{S} = \{S_l\}_{l \in \{1,\ldots,12\}}$
- Cell configuration CC of i and its neighbors j_1, j_2, j_3:

$$\mathbf{l} = (l, l_1, l_2, l_3) = (1, 9, 7, 12) \in \{1, \ldots, 12\}^4$$

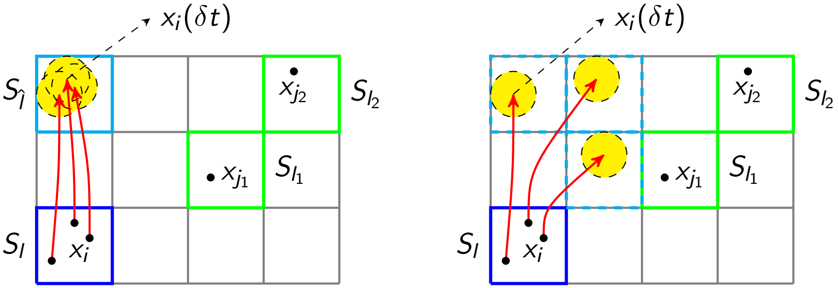
- Cell decomposition diameter: $d_{\text{max}} = \sqrt{2}$

Well Posed Discretizations

Given the cell decomposition $\mathcal{S} = \{S_l\}_{l \in \{\mathcal{I}\}}$ and the time step δt, we say that the space-time discretization $\mathcal{S}$-δt is well posed if for each $i = 1, \dots, N$ and CC $\mathbf{l}_i = (l_i, l_{j_1}, \dots, l_{j_{N_i}})$ of i

- there exists (at least one) cell $S_{l'_i}$
- and a control law assigned to the input v_i, such that for each $x_i(0) \in S_l$ and irrespectively of v_k, $k \neq i$ and the exact initial positions of the neighbors $x_{j_k}(0)$ in $S_{l_{j_k}}$
- agent i is driven to cell $S_{l'_i}$ exactly in time δt

Well Posed Discretizations



Sys. (A): $\dot{x}_i = f_{iA}(x_i, x_{j_1}, x_{j_2}) + v_{iA}$

Sys. (B): $\dot{x}_i = f_{iB}(x_i, x_{j_1}, x_{j_2}) + v_{iB}$

- The discretization is well posed for System (A)
- The discretization is not well posed for System (B)

Dynamics Properties[3]

- Lipschitz constants L_1, L_2

$$|f_i(x_i, \mathbf{x}_j) - f_i(x_i, \mathbf{y}_j)| \leq L_1 |\mathbf{x}_j - \mathbf{y}_j|$$

$$|f_i(x_i, \mathbf{x}_j) - f_i(y_i, \mathbf{x}_j)| \leq L_2 |x_i - y_i|$$

- Dynamics bounds

$$|f_i(x_i, \mathbf{x}_j)| \leq M$$

$$|v_i(t)| \leq v_{\max} \quad (< M)$$

$$\mathbf{x}_j := (x_{j_1}, \ldots, x_{j_{|\mathcal{N}_i|}})$$

[3] D. Boskos and D. V. Dimarogonas, Robust Connectivity Analysis for Multi-Agent Systems, CDC 2015

Analytical Results on Well Posed d_{max} - δt

QUESTION

- How do we quantify acceptable $d_{\max}$ - δt?

RESULT: Assuming that $v_{max} < M$, a sufficient condition which guarantees that the space-time discretization $d_{\max}$-δt is well posed, is that $d_{\max}$ and δt satisfy the following restrictions

$$
\begin{aligned}
d_{\max} &\in \left(0, \frac{v_{\max}^2}{4ML}\right] \\
\delta t &\in \left[\frac{v_{\max} - \sqrt{v_{\max}^2 - 4MLd_{\max}}}{2ML}, \frac{v_{\max} + \sqrt{v_{\max}^2 - 4MLd_{\max}}}{2ML}\right]
\end{aligned}
$$

with the dynamics dependent parameter L defined as

$$L := \max\{2L_2 + 4L_1\sqrt{N_i}, i = 1, \ldots, N\}$$

Analytical Results on Well Posed d_{max} - δt

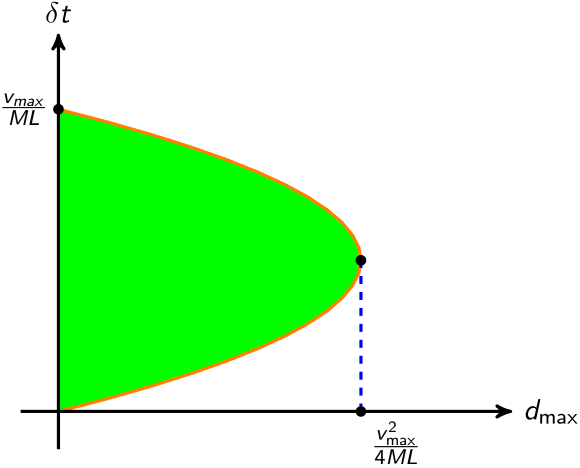


Figure: *Feasible $d_{\max} - \delta t$ region*

Selection of d_{max} - δt for Motion Planning

Transition possibilities can be quantified by employing additional d.o.f.!

PROPOSITION

Consider a cell decomposition $\mathcal{S}$ of D with diameter $d_{\max}$, a time step δt, the parameters $\lambda \in (0, 1)$, $\mu > 0$ and define

$$r := \lambda v_{\max} \delta t$$

We assume that r satisfies the design requirement

$$r \geq \frac{\mu}{2} d_{\max}$$

Then the space-time discretization is well posed for the multi-agent system, provided that λ, μ, $d_{\max}$ and δt satisfy certain algebraic sufficient conditions.

Selection of d_{max} - δt for Motion Planning

COROLLARY

Consider a cell decomposition $\mathcal{S}$ with diameter $d_{\max}$, a time step δt, and parameters $\lambda \in (0,1)$, $\mu > 0$ such that the hypotheses above are fulfilled. Then for each agent $i \in \{1, \ldots, N\}$ and each CC of i, there exist at least

$$\begin{aligned} \lfloor \mu^n \rfloor + 1, \text{ if } \mu^n \notin \mathbb{N}, \\ \lfloor \mu^n \rfloor, \text{ if } \mu^n \in \mathbb{N}, \end{aligned}$$

possible discrete transitions.

Corresponding Transition System

Agent's i individual transition system $TS_i := (Q, Act_i, \longrightarrow_i)$

- state set Q the indices $\mathcal{I}$ of the cell decomposition
- actions all possible cell indices of i and its neighbors

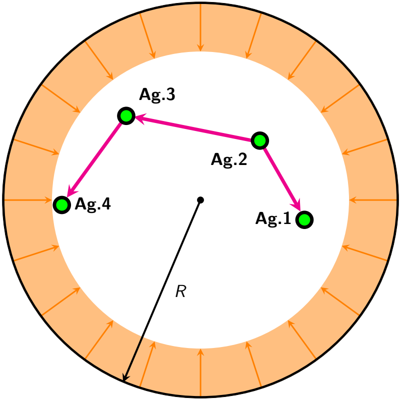
$$Act_i := \mathcal{I}^{N_i+1}$$

  (the set of all possible cell configurations of i)

- transition relation $\longrightarrow_i \subset Q \times Act_i \times Q$ as follows: For $l_i, l_i' \in Q$ and $\mathbf{l}_i = (l_i, l_{j_1}, \ldots, l_{j_{N_i}}) \in \mathcal{I}^{N_i+1}$,

$$l_i \xrightarrow{\mathbf{l}_i}_i l_i' \quad \text{iff} \quad l_i \xrightarrow{\mathbf{l}_i} l_i' \quad \text{is well posed.}$$

Example with Four Agents



- Network topology $\mathcal{N}_1 = \{2\}$, $\mathcal{N}_2 = \emptyset$, $\mathcal{N}_3 = \{2\}$, $\mathcal{N}_4 = \{3\}$
- Bounded circular domain of radius R
- Connectivity distance between neighboring agents ρ

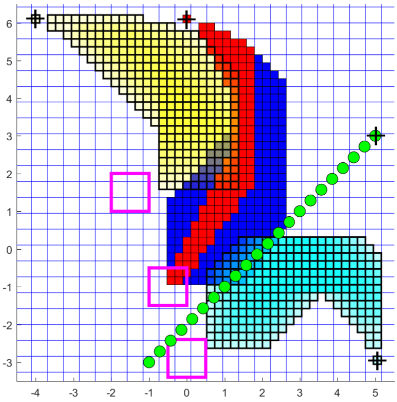
Dynamics and Selection of v_{max}

- Saturated dynamics

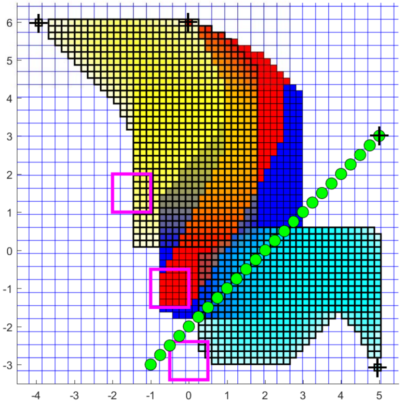
$$
\begin{aligned}
\dot{x}_1 &= \mathrm{sat}_\rho(x_2 - x_1) + g(x_1) + v_1 \\
\dot{x}_2 &= g(x_2) + v_2 \\
\dot{x}_3 &= \mathrm{sat}_\rho(x_2 - x_3) + g(x_3) + v_3 \\
\dot{x}_4 &= \mathrm{sat}_\rho(x_3 - x_4) + g(x_4) + v_4
\end{aligned}
$$

- $\mathrm{sat}_\rho(x) := x$ if $|x| \leq \rho$; $\mathrm{sat}_\rho(x) := \frac{\rho}{|x|}x$, if $|x| > \rho$
- Repulsion vector filed $g(x)$
- Selecting $v_{\mathrm{max}} = \frac{\rho}{2}$ ensures that initially connected configurations remain connected

Simulation Results



(i)



(ii)

- Reachable cells: (i) $\lambda = 0.2$ and (ii) $\lambda = 0.3$
- Agents: 1-cyan, 2-green, 3-blue, 4-yellow
- Agent 4 reaches its target box with the finer discretization, also due to the increased number of (red) paths of 3 that reach its target box

Ongoing and Future Work

- Abstractions of varying decentralization degree[4]
    - based on discrete positions up to a distance in the network graph
    - improved discretizations due to the reduction of the required control for the coupling terms
- Online abstractions
    - based on the discretization of each agent's reachable set over a time horizon
    - applicable to forward complete systems
    - improved discretizations and reachability properties for agents with weaker couplings over the horizon
- Future directions: higher order systems, special network structures ...

[4] D. Boskos and D. V. Dimarogonas, Abstractions of Varying Decentralization Degree for Coupled Multi-Agent Systems, CDC 2016

Table of Contents

Aim

- A team $\mathcal{N} = \{1, \ldots, N\}$ of agents
  - A finite discrete transition system $\mathcal{T}_i$
    - Abstraction of action capabilities
    - Example: transition system emerging from previous abstraction procedure
  - Synchronization capabilities
- High-level behavior specification
  - *Motion* LTL specification ϕ_i over the states
  - *Task* LTL specification ψ_i over the inputs/actions
- Efficiently synthesize controllers fulfilling the tasks
  - A *satisfying* trace of each $\mathcal{T}_i$
  - Necessary synchronizations
  - The catch: dependencies at the task (discrete) level

Problem Formulation

For each $i \in \mathcal{N}$, synthesize appropriate motion and action sequences so that

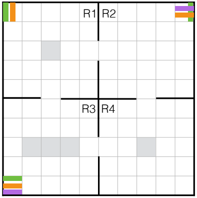
- the set of induced behaviors is nonempty
- the motion specification ϕ_i is satisfied
- the task specification ψ_i is locally satisfied

Example I

Agent 1 is a ground vehicle and has to avoid walls and obstacles.
Agent 2 and Agent 3 are UAVs and their environment is obstacle-free except for the walls.

Motion specifications

Agent 1: Keep avoiding R1, $\phi_1 = \mathcal{G}\neg \mathrm{R1}_1$.
Agent 2: Keep avoiding R2, $\phi_2 = \mathcal{G}\neg \mathrm{R2}_2$.
Agent 3: Periodically survey R1 and R2, $\phi_3 = \mathcal{G}\,\mathcal{F}\,\mathrm{R1}_3 \wedge \mathcal{G}\,\mathcal{F}\,\mathrm{R2}_3$.



Task specifications

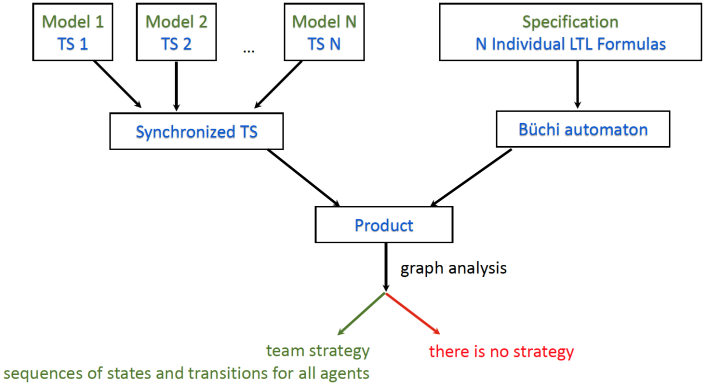
Agent 1: periodically *load*(–) with the *help* of agent 2 (–) and the *assistance* of agent 3 (–), then *unload* (|) with the *help* of agent 2 (–) or the *assistance* of agent 3 (–)

$$\psi_1 = \mathit{load} \wedge \mathit{help} \wedge \mathit{assist} \wedge \mathcal{G}\,(\mathit{load} \Rightarrow \mathcal{X}\,(\mathit{unload} \wedge (\mathit{help} \vee \mathit{assist}))) \wedge \mathcal{G}\,(\mathit{unload} \Rightarrow \mathcal{X}\,(\mathit{load} \wedge \mathit{help} \wedge \mathit{assist}))$$

Agent 2: Periodically provide *inform* service (|), $\psi_2 = \mathcal{G}\mathcal{F}\mathit{inform}$.

Agent 3: Nothing specific, $\psi_3 = \mathit{true}$.

Straightforward Approach



Computational infeasibility!

Our Hierarchical Approach I

- Each ϕ_i is translated to a Büchi automaton $\mathcal{B}_i^{\phi}$
- N motion products $\mathcal{P}_i = \mathcal{T}_i \otimes \mathcal{B}_i^{\phi}$ are built
- Each motion product is reduced to $\ddot{\mathcal{P}}_i$ by systematic removal of states, where no services of interest are available
- Each ψ_i is translated to a Büchi automaton $\mathcal{B}_i^{\psi}$
- N task and motion products $\bar{\mathcal{P}}_i = \ddot{\mathcal{P}}_i \otimes \mathcal{B}_i^{\psi}$
- Each motion and task product is reduced to $\widehat{\mathcal{P}}_i$ by systematic removal of states, where no dependent services are available
- A global product $\mathcal{P} = \widehat{\mathcal{P}}_1 \otimes \ldots \otimes \widehat{\mathcal{P}}_N$ containing only states relevant for planning of dependent tasks is constructed

Our Hierarchical Approach II

- An accepting run in the global product projected onto the original system gives
  - a motion plan
  - a task execution plan
  - a synchronization plan

  for each agent i, that is *correct-by-design with respect to ϕ_i and ψ_i.*

Example I Revisited

Agent 1 is a ground vehicle and has to avoid walls and obstacles.
Agent 2 and Agent 3 are UAVs and their environment is obstacle-free except for the walls.

Motion specifications

Agent 1: Keep avoiding R1, $\phi_1 = \mathcal{G}\neg \mathrm{R1}_1$.
Agent 2: Keep avoiding R2, $\phi_2 = \mathcal{G}\neg \mathrm{R2}_2$.
Agent 3: Periodically survey R1 and R2, $\phi_3 = \mathcal{G}\,\mathcal{F}\,\mathrm{R1}_3 \wedge \mathcal{G}\,\mathcal{F}\,\mathrm{R2}_3$.

Task specifications

Agent 1: periodically *load*(–) with the *help* of agent 2 (–) and the *assistance* of agent 3 (–), then *unload* (|) with the *help* of agent 2 (–) or the *assistance* of agent 3 (–)

$$\psi_1 = \mathit{load} \wedge \mathit{help} \wedge \mathit{assist} \wedge \mathcal{G}\,(\mathit{load} \Rightarrow \mathcal{X}\,(\mathit{unload} \wedge (\mathit{help} \vee \mathit{assist}))) \wedge \mathcal{G}\,(\mathit{unload} \Rightarrow \mathcal{X}\,(\mathit{load} \wedge \mathit{help} \wedge \mathit{assist}))$$

Agent 2: Periodically provide *inform* service (|), $\psi_2 = \mathcal{G}\mathcal{F}\mathit{inform}$.
Agent 3: Nothing specific, $\psi_3 = \mathit{true}$.

Example I Revisited

Centralized approach

- Each TS: 100 states
- Product TS: 100^3 states
- $\mathcal{B}_1^\phi, \mathcal{B}_2^\phi, \mathcal{B}_3^\phi, \mathcal{B}_1^\psi, \mathcal{B}_2^\psi, \mathcal{B}_3^\psi$: $2, 2, 3, 2, 2, 1$ states, respectively respectively
- Intersection BA: $2 \cdot 2 \cdot 3 \cdot 2 \cdot 2 \cdot 1 \cdot 7 = 330$ states
- The overall product $\mathcal{P}$: ≈ 330 mil. states

Our approach:

- $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$: $200, 200, 300$ states, respectively
- $\widehat{P}_1, \widehat{P}_2, \widehat{P}_3$: 27,17,8 states, respectively
- The largest structure handled has cca 15000 states.

Remarks

- Worst-case complexity meets the complexity of the centralized solution
- Suitable for sparsely distributed services of interest and occasional needs for collaboration
- The bottleneck is still the product $\mathcal{P}$ and (some) synchronization
- Extension to event-based receding horizon approach: uses local versions of product and synchronizations in an event-based fashion

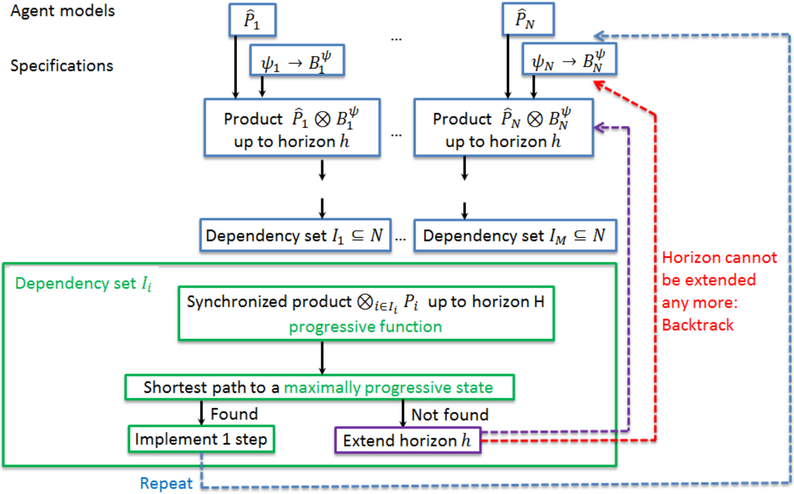
Event-triggered Receding Horizon Approach

- Each ϕ_i is translated to a Büchi automaton $\mathcal{B}_i^{\phi}$
- N motion products $\mathcal{P}_i = \mathcal{T}_i \otimes \mathcal{B}_i^{\phi}$ are built
- Each motion product is reduced to $\ddot{\mathcal{P}}_i$ by systematic removal of states, where no services of interest are available
- Each ψ_i is translated to a Büchi automaton $\mathcal{B}_i^{\psi}$
- N task and motion products $\bar{\mathcal{P}}_i = \ddot{\mathcal{P}}_i \otimes \mathcal{B}_i^{\psi}$
- Each motion and task product is reduced to $\widehat{\mathcal{P}}_i$ by systematic removal of states, where no dependent services are available
- A global product $\mathcal{P} = \widehat{\mathcal{P}}_1 \otimes \ldots \otimes \widehat{\mathcal{P}}_N$ containing only states relevant for planning of dependent tasks is constructed

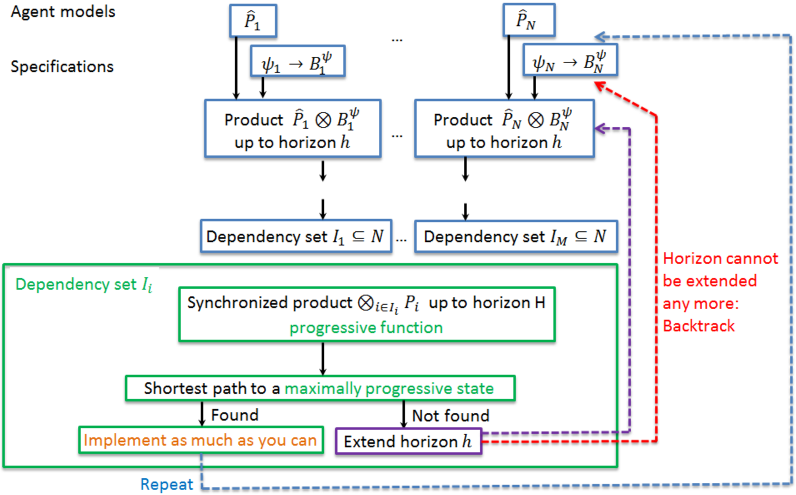
Event-triggered Receding Horizon Approach

- Translate the infinite-horizon problem into an infinite sequence of finite-horizon problems
- Dynamically partition the agents based on dependency
- Define progressive function to indicate closeness to goal satisfaction
- Introduce event-triggered synchronization

Stepwise Receding Horizon



Stepwise Receding Horizon



Example II

- Agent 1 can load (l_H, l_A, l_B), carry, and unload (u_H, u_A, u_B) a heavy object H or a light object A, B, in the green cells.

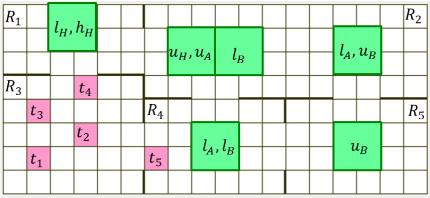
$$\psi_1 = \mathcal{F}(l_H \wedge h_H \wedge \mathcal{X}\, u_H \wedge \bigwedge_{i \in \{A,B\}} \mathcal{G}\mathcal{F}\,(l_i \wedge \mathcal{X} u_i)))$$

- Agent 2 is capable of helping the agent 1 to load object H (h_H), and to execute simple tasks in the purple regions $(t_1 - t_5)$.

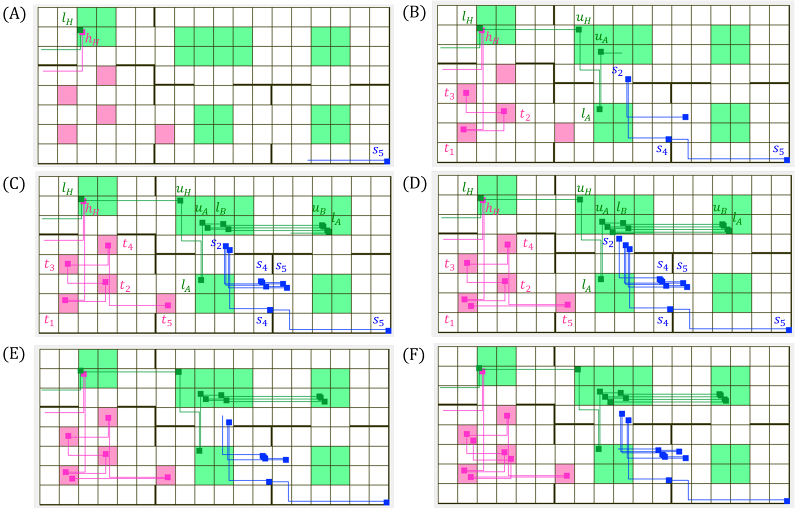
$$\psi_2 = \mathcal{G}\mathcal{F}\ (t_1 \wedge \mathcal{X}\ (t_2 \wedge \mathcal{X}\ (t_3 \wedge \mathcal{X}\ (t_4 \wedge \mathcal{X}\ \ t_5 \wedge s_4))))))$$

- Agent 3 is capable of taking a snapshot of the rooms $(s_1 - s_5)$ when being present in there.

$$\psi_3 = \bigwedge_{i \in \{2,4,5\}} \mathcal{G}\mathcal{F}\, s_i$$



Example II



cca 3 mil. vs. hundreds to thousands of states

Remarks

- The worst-case complexity still the same as for the centralized case
- Suitable for collaborations executed in small (dynamically changing) subgroups

Conclusion and Future Work

- Conclusion
  - Decentralized abstractions and planning for multi-agent systems
  - Consideration of dynamics and continuous-time constraints
  - Decomposition of formulas and event-based horizon framework for decentralized LTL based planning
- Future and current Work
  - Further reduction of complexity in distributed task planning
  - More general dynamics and combination with dependent tasks
  - Online version of abstraction framework
  - Quantifying space and time constraints at the task level (MITL and STL specs)

References and acks

- First part: discrete specs and coupled constraints: Guo et al., CDC14-15, IJRR15, TAC16
- Second part: locally defined abstractions for MAS: Boskos and Dimarogonas, CDC15,CDC 16, TAC16 under revision
- Third part: distributed task planning: Tumova and Dimarogonas ACC14, Automatica16, CDC15
- EU Projects: ERC StG BUCOPHSYS, FP7 RECONFIG, H2020 AEROWORKS and Co4Robots. National projects: VR, SSF, KAW IPSYS, KAW Fellowship.
- Contact: `http://people.kth.se/~dimos/`