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

OptHySYS Workshop, Trento, January 2017

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

Automatic Control,
KTH Royal Institute of Technology, Sweden
Table of Contents

Introduction

Multi-Agent Hybrid Control under Local LTL Tasks and Relative-Distance Constraints

Abstractions for Constrained Multi-Agent Systems

Multi-Agent Planning from Local LTL Specifications
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 representations (aka abstractions) of control systems
State of the Art

Single Agent-Single Task

- 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

Introduction

Multi-Agent Hybrid Control under Local LTL Tasks and Relative-Distance Constraints

Abstractions for Constrained Multi-Agent Systems

Multi-Agent Planning from Local LTL Specifications
Problem Formulation

- A team of $N$ mobile agents, $x_i(t), u_i(t) \in \mathbb{R}^2$:
  \[ \dot{x}_i(t) = u_i(t), \quad i \in \mathcal{N} = \{1, \cdots, N\}. \]

- 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 $\Sigma_i$ available at each region in $\mathcal{R}_i$. 
Problem Formulation, cont’d

- **Local LTL task** specification $\varphi_i$, over $\Sigma_i$.

- Note that $\varphi_i$ can be co-safe or general LTL formulas.

- $\varphi_i$ 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 $S_i$ such that

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

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 $\varphi_i$ and minimizes a cost function.

- **Automata-based** model-checking algorithm\(^1\)
- Discrete plan *synthesized locally* by each agent $i \in \mathcal{N}$:

$$S_i = \sigma_{i1} \cdots \sigma_{i\tilde{s}_i} (\sigma_i(s_{i+1}) \cdots \sigma_i N_i) ^\omega, \quad \sigma_{is_i} = (R_{is_i}, \Sigma_{is_i}).$$

- Our algorithm minimizes the **maximal distance** between two consecutive regions along the plan\(^2\).

---


Step 2. 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:
  \[ C_{\text{act}} : \quad u_i(t) \triangleq -d_i \, p_i - \sum_{j \in \mathcal{N}_i(t)} h_{ij} \, x_{ij}, \]

  (2) the *passive* mode:
  \[ C_{\text{pas}} : \quad u_i(t) \triangleq - \sum_{j \in \mathcal{N}_i(t)} h_{ij} \, x_{ij}, \]

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

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

  • \( \varepsilon > 0 \) 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 $i^* \in \mathcal{N}_a$, such that $x_j(T_f) \in R_{i^*g}$, $\forall j \in \mathcal{N}$.
- All agents will enter $R_{i^*g}$, i.e., $x_j \in R_{i^*g}$, $\forall j \in \mathcal{N}$.
- The above holds for any number of active agents that $1 \leq \mathcal{N}_a \leq \mathcal{N}$.
Potential-field-based Design

Consider the following potential-field function:

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

- \( \phi_c(x_{ij}) \) 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:

  $$S_i \triangleq \{ x \in \mathbb{R}^{2N} \mid \|x - 1_N \otimes c_{ig}\| \leq r_S(\varepsilon) \}, \quad \forall i \in \mathcal{N}_a.$$ 

  Let $S \triangleq \bigcup_{i \in \mathcal{N}_a} S_i$ and $S^- \triangleq \mathbb{R}^{2N} \setminus S$.

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

- **Lemma 2,3**: There exists $\varepsilon < \min\{\varepsilon_2, \varepsilon_6\}$ such that regions $\{S_i\}$ are sufficiently far. Critical points are close to the region center.
Lemma 4: There exists $\varepsilon_{\text{min}} > 0$ such that if $\varepsilon < \varepsilon_{\text{min}}$, all critical points of $V$ within $S$ are local minima.

Convergence Results

There exists a finite time $T_f \in [T_s, \infty)$ and one active agent $i^* \in \mathcal{N}_a$, such that $x_j(T_f) \in R_{i^*g}$, $\forall j \in \mathcal{N}$, 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 $S_{i^*}$ for one active agent $i^* \in \mathcal{N}_a$.
- All agents would enter $R_{i^*g}$, i.e., $x_j \in R_{i^*g}$, $\forall j \in \mathcal{N}$.
- 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 $1 \leq N_a \leq N$. 
Step 3. Hybrid Control: sc-safe LTL task case

Case one
All tasks \( \{ \varphi_i \} \) are given as sc-safe LTL formulas.

- If \( \varphi_i \) 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 \( \Sigma_{ik} \) 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 \( \forall i \in \mathcal{N}, \varphi_i \) is eventually satisfied, and
  \[\|x_i(t) - x_j(t)\| < r, \forall (i, j) \in E(0) \text{ and } \forall t \geq 0.\]
Step 3. 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

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

Sc-safe LTL task

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

General LTL task

- $\varphi_1 = \square \lozenge \sigma_{11} \land \square \lozenge \sigma_{12}$.
- $\varphi_2 = \square \lozenge (\sigma_{21} \lor \sigma_{22} \lor \sigma_{23})$.
- $\varphi_3 = \square \lozenge (\sigma_{31} \lor \sigma_{32} \lor \sigma_{33})$.
- $\varphi_4 = \square \lozenge \sigma_{41} \land \square \lozenge \sigma_{42}$.
Scenario one
Scenario two
# Table of Contents

Introduction

Multi-Agent Hybrid Control under Local LTL Tasks and Relative-Distance Constraints

Abstractions for Constrained Multi-Agent Systems

Multi-Agent Planning from Local LTL Specifications
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

\[ \dot{x}_i = u_i = f_i(x_i, x_j) + v_i, \ x_j = (x_{j1}, \ldots, x_{jN_i}), \ i = 1, \ldots, N \]

- Closed loop system with coupled constraints \( f_i(x_i, x_j) \) 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 → finite or countable “partition”
  \[ S = \{ S_l \}_{l \in I} \]
  of the workspace by uniformly bounded sets
  • time step \( \delta t \)
  • which ensure that the discretized model of closed loop system
    is well posed - meaningful

• Notation

  • Cell Configuration CC of \( i \) and its neighbors \( j_1, \ldots, j_{N_i} \)
  \[ N_i + 1 \text{-tuple of cell indices } l_i = (l_i, l_{j_1}, \ldots, l_{j_{N_i}}) \in I^{N_i+1} \]

• Cell decomposition diameter \( d_{max} \):
  • “maximum” diameter of a cell \( S_l \in S \)
  \[ d_{max} := \sup \{|x - y| : x, y \in S_l, l \in I\} \]
Cell Decomposition - Cell Configuration Example

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

\[
I = (I, I_1, I_2, I_3) = (1, 9, 7, 12) \in \{1, \ldots, 12\}^4
\]

- **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 \( \delta t \), we say that the space-time discretization \( S-\delta t \) is well posed if for each \( i = 1, \ldots, N \) and \( \text{CC } \mathbf{l}_i = (l_i, l_{j_1}, \ldots, l_{j_{\mathbf{N}_i}}) \) of \( i \)

- there exists (at least one) cell \( S_{i'} \)

- 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 \( x_{jk}(0) \) in \( S_{l_{jk}} \)

- agent \( i \) is driven to cell \( S_{i'} \) exactly in time \( \delta t \)
Well Posed Discretizations

Sys. (A): \( \dot{x}_i = f_{iA}(x_i, x_{j1}, x_{j2}) + v_{iA} \)

Sys. (B): \( \dot{x}_i = f_{iB}(x_i, x_{j1}, x_{j2}) + v_{iB} \)

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

- **Lipschitz constants** $L_1, L_2$

  $$|f_i(x_i, x_j) - f_i(x_i, y_j)| \leq L_1|x_j - y_j|$$

  $$|f_i(x_i, x_j) - f_i(y_i, x_j)| \leq L_2|x_i - y_i|$$

- **Dynamics bounds**

  $$|f_i(x_i, x_j)| \leq M$$

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

  $$x_j := (x_{j1}, \ldots, x_{j|N_i|})$$

---

$^3$D. Boskos and D. V. Dimarogonas, Robust Connectivity Analysis for Multi-Agent Systems, CDC 2015
Analytical Results on Well Posed $d_{\text{max}} - \delta t$

QUESTION

- How do we quantify acceptable $d_{\text{max}} - \delta t$?

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

$$d_{\text{max}} \in \left(0, \frac{v_{\text{max}}^2}{4ML}\right]$$

$$\delta t \in \left[\frac{v_{\text{max}} - \sqrt{v_{\text{max}}^2 - 4MLd_{\text{max}}}}{2ML}, \frac{v_{\text{max}} + \sqrt{v_{\text{max}}^2 - 4MLd_{\text{max}}}}{2ML}\right]$$

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 $d_{\text{max}} - \delta t$

Figure: Feasible $d_{\text{max}} - \delta t$ region
Selection of $d_{\text{max}} - \delta t$ for Motion Planning

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

**PROPOSITION**

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

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

We assume that $r$ satisfies the design requirement

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

Then the space-time discretization is well posed for the multi-agent system, provided that $\lambda$, $\mu$, $d_{\text{max}}$ and $\delta t$ satisfy certain algebraic sufficient conditions.
COROLLARY
Consider a cell decomposition $\mathcal{S}$ with diameter $d_{\text{max}}$, a time step $\delta t$, 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

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

$$\lfloor \mu^n \rfloor, \text{ if } \mu^n \in \mathbb{N},$$

possible discrete transitions.
Agent’s $i$ individual transition system $TS_i := (Q, Act_i, \rightarrow_i)$

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

$$l_i \xrightarrow{l_i}{\rightarrow_i} l'_i \quad \text{iff} \quad l_i \xrightarrow{l_i}{\rightarrow} l'_i \quad \text{is well posed}.$$
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 $\rho$
Dynamics and Selection of $v_{max}$

- Saturated dynamics

\[
\begin{align*}
\dot{x}_1 &= \text{sat}_\rho(x_2 - x_1) + g(x_1) + v_1 \\
\dot{x}_2 &= g(x_2) + v_2 \\
\dot{x}_3 &= \text{sat}_\rho(x_2 - x_3) + g(x_3) + v_3 \\
\dot{x}_4 &= \text{sat}_\rho(x_3 - x_4) + g(x_4) + v_4
\end{align*}
\]

- $\text{sat}_\rho(x) := x$ if $|x| \leq \rho$; $\text{sat}_\rho(x) := \frac{\rho}{|x|}x$, if $|x| > \rho$

- Repulsion vector filed $g(x)$

- Selecting $v_{\text{max}} = \frac{\rho}{2}$ ensures that initially connected configurations remain connected
Simulation Results

- 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

Introduction

Multi-Agent Hybrid Control under Local LTL Tasks and Relative-Distance Constraints

Abstractions for Constrained Multi-Agent Systems

Multi-Agent Planning from Local LTL Specifications
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 $\phi_i$ over the states
  • $Task$ LTL specification $\psi_i$ 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 $\phi_i$ is satisfied
- the task specification $\psi_i$ is locally satisfied
Example 1

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, \( \phi_1 = G \neg R_1 \).
Agent 2: Keep avoiding R2, \( \phi_2 = G \neg R_2 \).
Agent 3: Periodically survey R1 and R2, \( \phi_3 = GF R_1 \land GF R_2 \).

Task specifications
Agent 1: periodically load\( (\neg) \) with the help of agent 2 \( (\neg) \) and the assistance of agent 3 \( (\neg) \), then unload \( (|) \) with the help of agent 2 \( (\neg) \) or the assistance of agent 3 \( (\neg) \)

\[
\psi_1 = load \land help \land assist \land G (load \Rightarrow X (unload \land (help \lor assist))) \land G (unload \Rightarrow X (load \land help \land assist))
\]

Agent 2: Periodically provide inform service \( (|) \), \( \psi_2 = GF inform \).
Agent 3: Nothing specific, \( \psi_3 = true \).
Straightforward Approach

Model 1
  TS 1

Model 2
  TS 2

...  ...

Model N
  TS N

Synchronized TS

Product

Specification
  N Individual LTL Formulas

Büchi automaton

graph analysis

team strategy
sequences of states and transitions for all agents

there is no strategy

Computational infeasibility!
Our Hierarchical Approach I

- Each $\phi_i$ is translated to a Büchi automaton $B^\phi_i$
- $N$ motion products $P_i = T_i \otimes B^\phi_i$ are built
- Each motion product is reduced to $\bar{P}_i$ by systematic removal of states, where no services of interest are available
- Each $\psi_i$ is translated to a Büchi automaton $B^\psi_i$
- $N$ task and motion products $\bar{P}_i = \bar{P}_i \otimes B^\psi_i$
- Each motion and task product is reduced to $\hat{P}_i$ by systematic removal of states, where no dependent services are available
- A global product $P = \hat{P}_1 \otimes \ldots \otimes \hat{P}_N$ 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 $\phi_i$ and $\psi_i$. 
Example 1 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, $\phi_1 = G \neg R_1$.

**Agent 2:** Keep avoiding R2, $\phi_2 = G \neg R_2$.

**Agent 3:** Periodically survey R1 and R2, $\phi_3 = G F R_1 \land G F R_2$.

**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 (−)

$$\psi_1 = load \land help \land assist \land G (load \Rightarrow X (unload \land (help \lor assist))) \land G (unload \Rightarrow X (load \land help \land assist))$$

**Agent 2:** Periodically provide inform service (︱), $\psi_2 = G F inform$.

**Agent 3:** Nothing specific, $\psi_3 = true$. 

Example I Revisited

Centralized approach

- Each TS: 100 states
- Product TS: $100^3$ states
- $B_1^\phi, B_2^\phi, B_3^\phi, B_1^\psi, B_2^\psi, B_3^\psi$: 2, 2, 3, 2, 2, 1 states, respectively
- Intersection BA: $2 \cdot 2 \cdot 3 \cdot 2 \cdot 2 \cdot 1 \cdot 7 = 330$ states
- The overall product $P$: $\approx 330$ mil. states

Our approach:

- $P_1, P_2, P_3$: 200, 200, 300 states, respectively
- $\hat{P}_1, \hat{P}_2, \hat{P}_3$: 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 $\phi_i$ is translated to a Büchi automaton $B_i^\phi$
- $N$ motion products $P_i = T_i \otimes B_i^\phi$ are built
- Each motion product is reduced to $\bar{P}_i$ by systematic removal of states, where no services of interest are available
- Each $\psi_i$ is translated to a Büchi automaton $B_i^\psi$
- $N$ task and motion products $\bar{P}_i = \bar{P}_i \otimes B_i^\psi$
- Each motion and task product is reduced to $\hat{P}_i$ by systematic removal of states, where no dependent services are available
- A global product $P = \hat{P}_1 \otimes \ldots \otimes \hat{P}_N$ 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

Agent models

Specifications

Dependency set $I_i \subseteq N$

Synchronized product $\bigotimes_{i \in I_i} P_i$ up to horizon $H$

progressive function

Shortest path to a maximally progressive state

Found

Implement 1 step

Not found

Extend horizon $h$

Repeat

Horizon cannot be extended any more:

Backtrack
Stepwise Receding Horizon

Agent models

Specifications

Dependency set $I_1 \subseteq N$

Product $\hat{P}_1 \otimes B_1^\psi$ up to horizon $h$

Dependency set $I_M \subseteq N$

Product $\hat{P}_N \otimes B_N^\psi$ up to horizon $h$

Synchronized product $\bigotimes_{i \in I_t} P_i$ up to horizon $H$

Progressive function

Shortest path to a maximally progressive state

Found

Implement as much as you can

Not found

Extend horizon $h$

Repeat

Horizon cannot be extended any more: Backtrack
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.

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

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

\[
\psi_2 = \mathcal{G}\mathcal{F} (t_1 \land \mathcal{X} (t_2 \land \mathcal{X} (t_3 \land \mathcal{X} (t_4 \land \mathcal{X} t_5 \land s_4))))
\]

• 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, TAC17
- Second part: locally defined abstractions for MAS: Boskos and Dimarogonas, CDC15, CDC16, SIAM17
- Third part: distributed task planning: Tumova and Dimarogonas ACC14, Automatica16, CDC15
- Contact: http://people.kth.se/~dimos/
Last slide

Grazie!
Case two: general LTL task

Case two

All tasks \( \{ \varphi_i \} \) are given as general LTL formulas.

- If \( \varphi_i \) is general, every agent has an infinite plan

\[
\tau_i = (R_{i1}, \Sigma_{i1}) \cdots [(R_{iK_i}, \cdots \Sigma_{iK_i}) \cdots (R_{iN_i}, \Sigma_{iN_i})]^{\omega}
\]

- The previous approach may not work.

- **Round**: time interval \( [T_{\diamond m-1}, T_{\diamond m}) \) when every agent has made a progress in executing its plan \( \tau_i \).

- **Reaching-event detector**: \( \Omega_i(j, t) = \text{True} \) if agent \( i \) detects that agent \( j \) reaches \( R_{jg} \) at time \( t \).
• Local variables: \( \chi_i \geq 0, \quad \Upsilon_i \in \mathbb{Z}^N \),

Local switching iterative policy (from agent \( i \)'s view)

(I) State in plan \( (R_{i\kappa_i}, \Sigma_{i\kappa_i}) \), where \( \kappa_i := 1; \quad \chi_i := 0; \quad \Upsilon_i := 0^N \).

(II) If agent \( i \) reaches \( R_{i\kappa_i} \), then provide services \( \Sigma_{i\kappa_i} \). Set \( \kappa_i := \kappa_i + 1 \) and \( \Upsilon_i[j] := \Upsilon_i[j] + 1 \).

• Stay active \( (b_i = 1) \) or become passive \( (b_i = 0) \) based on the progress so far within the current round.
• Maximal number of progresses allowed.

(III) If \( \Omega_i(j, t) = \text{True} \), set \( \Upsilon_i[j] := \Upsilon_i[j] + 1 \).

(IV) Whenever \( \Upsilon_i[j] \geq 1, \forall j \in \mathcal{N} \), set \( \Upsilon_i := 0^N, \chi_i := t \).

• The round \( [T_{\bigcirc m-1}, T_{\bigcirc m}) \) is finite, \( \forall m \geq 1 \).
• Guaranteed that \( \forall i \in \mathcal{N}, \varphi_i \) is eventually satisfied, and \( \| x_i(t) - x_j(t) \| < r, \forall (i, j) \in E(0) \) and \( \forall t \geq 0 \).
Case three: mixed task

Case three

Any task $\varphi_i$ can be a either sc-safe or general LTL formula.

- $\mathcal{N} = \mathcal{N}_{ge} \cup \mathcal{N}_{sc}$.
- All-passive detector, to detect agents with sc-safe tasks.
- Similar switching policy as before, but excluding $\mathcal{N}_{sc}$ when evaluating $\Upsilon_i$. 