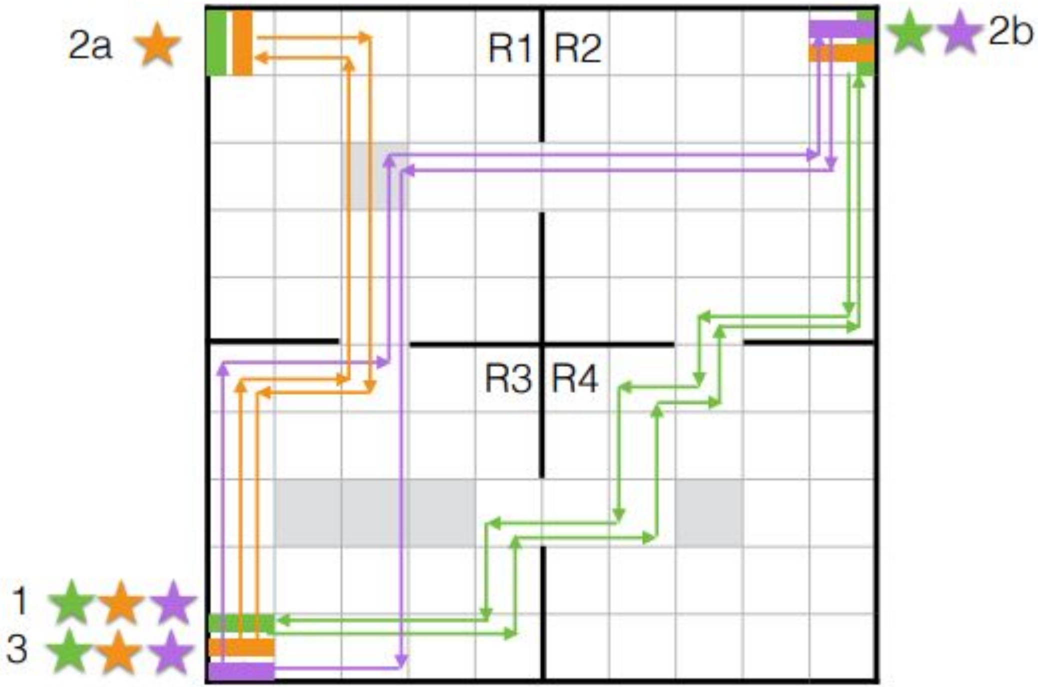
Multi-Agent Cooperative Planning

Aadil Hayat
Rajeev Kumar
Anup Nayak

Cooperative Planning Problem



Agent 1 (Green) :
Land Vehicle
Services: *load* and *unload*

Agent 2 (Orange) :
UAV
Services: *help* and *inform*

Agent 3 (Purple) :
UAV
Services: *assist*

Decomposition of Multi-Agent Planning under Distributed Motion and Task LTL Specifications

Jana Tumova and Dimos V. Dimarogonas

Overview

- Efficient procedure for discrete multi-agent planning under LTL specifications
- Two-phase automata-based solution
- Decoupling the planning procedure for two types of specifications
- First Step: fully decentralized satisfaction of motion specifications
- Second Step: synchronized planning over abstracted representation of admissible traces

Notations and Preliminaries

- Labeled Transition System

A *labeled transition system (TS)* is a tuple $\mathcal{T} = (S, s_{init}, A, T, \Pi, L)$, where S is a finite set of states; $s_{init} \in S$ is the initial state; A is a finite set of actions; $T \subseteq S \times A \to S$ is a partial deterministic transition function; Π is a set of atomic propositions; and $L : S \to 2^{\Pi}$ is a labeling function. A *trace* of $\mathcal{T}$ is an infinite alternating sequence of states and actions $\tau = s_1\alpha_1 s_2\alpha_2 \ldots$, such that $s_1 = s_{init}$, and for all $i \geq 1$, $T(s_i, \alpha) = s_{i+1}$.

Notations and Preliminaries

- LTL formula

An *LTL formula* φ over the set of atomic propositions Σ is defined inductively: (i) $\pi \in \Sigma$ is a formula, and (ii) if φ_1 and φ_2 are formulas, then $\varphi_1 \vee \varphi_2$, $\neg\varphi_1$, $\mathsf{X}\,\varphi_1$, $\varphi_1 \,\mathsf{U}\, \varphi_2$, $\mathsf{F}\,\varphi_1$, and $\mathsf{G}\,\varphi_1$ are each a formula, where $\neg$ and $\vee$ are standard Boolean connectives, and X (next), U (until), F (eventually), and G (always) are temporal operators. The semantics of LTL are defined over infinite words over 2^{Π} (see, e.g., [1]). A trace $\tau = s_1\alpha_1 s_2\alpha_2 \ldots$ of $\mathcal{T}$ *satisfies* an LTL formula φ over Π ($\tau \models \varphi$) iff $L(s_1)L(s_2)\ldots$ satisfies φ ($L(s_1)L(s_2)\ldots \models \varphi$).

Problem Formulation

- Each agent’s temporal specification comprises of : a *motion* and a *task* formula

- Motion formula: safety, reachability, surveillance or sequencing

- Task formula: High-level task composed of simple services

- Motion formula is completely independent

- Task formula may involve collaboration of agents

Problem Formulation: System Model

- Action Execution Capabilities
  - Agent i action execution capabilities are modeled as a finite transition system

$$\mathcal{T}_i = (S_i, s_{init,i}, A_i, T_i, \Pi_i, L_i)$$

  - States of TS correspond to states of the agents
  - The atomic propositions represent inherent properties of the system states
  - The actions represent abstractions of the agent's low-level controllers
  - The traces are the abstractions of the agent's long-term behaviour
  - Each of the agent's action executions takes a certain amount of time
  - Transition duration is arbitrary and unknown prior to its execution

Problem Formulation: System Model

- Synchronization Capabilities
  - The agents have the ability to synchronize
  - They can wait for each other and proceed with the further execution simultaneously
  - When in state s, an agent i can send a request $sync_i(I)$ where $\{i\} \subseteq I \subseteq N$
  - Before proceeding it has to wait till $sync_{i'}(I)$ has been sent by each agent $i' \in I$
  - Assuming perfect propagation of the synchronization requests
  - The set of all synchronization requests of an agent i is $Sync_i$
  - Instead of allowing for idling, we include an existence of special action $stay_i$ called *self-loop*

Problem Formulation: System Model

- Services
    - Each agent's specification if given via two components
    - First one are temporal requirements on the atomic propositions that need to hold along its trace
    - Second one is a task given in terms of events of interest which we call *services*
    - The set of services that can be provided by an agent i is Σ_i
    - Services are provided within agents' transitions; each action is associated with a label

$$\mathcal{M}_i = (\mathcal{T}_i = (S_i, s_{init,i}, A_i, T_i, \Pi_i, L_i), Sync_i, \Sigma_i, \mathcal{L}_i)$$

Problem Formulation: System Model

- Behaviors

Definition 1 (Behavior and strategy). *A behavior of an agent i is a tuple $\beta = (\tau, \gamma, \mathbb{T})$, where $\tau = s_1\alpha_1 s_2\alpha_2 \ldots$ is a trace of $\mathcal{T}_i$; $\gamma = r_1 r_2 \ldots$ is a* synchronization sequence, *where $r_j \in Sync_i$ is the synchronization request sent at s_j; and $\mathbb{T} = t_{s_1} t_{\alpha_1} t_{s_2} t_{\alpha_2} \ldots$ is a non-decreasing* behavior time sequence, *where t_{s_j} is the time instant when the synchronization request r_j was sent, and t_{α_j} is the time instant when the action α_j started being executed. The following properties hold: $t_{s_1} = 0$, and for all $j \geq 1$, $t_{s_{j+1}} - t_{\alpha_j} = \Delta_{\alpha_j}$, and $t_{\alpha_j} - t_{s_j} = \Delta_{s_j}$. A* strategy *$(\tau, \gamma)$ for an agent i is a trace τ and a synchronization sequence γ.*

Problem Formulation: System Model

- Induced Behaviors

Definition 2 (Induced behaviors). *The set of behaviors induced by a collection of strategies* $(\tau_1, \gamma_1), \ldots, (\tau_N, \gamma_N)$ *of agents in* $\mathcal{N}$ *are the subset of the collections of their behaviors* $\mathbb{B} \subseteq \{\mathfrak{B} \in \{\beta_1, \ldots, \beta_N \mid \beta_i$ *is a behavior of agent* $i\}\}$ *satisfying the following condition for all* $i \in \mathcal{N}$, *and* $j \geq 1$: *Suppose that* $r_{i,j} = sync_i(I)$. *Then for all* $i' \in I$ *there exists a* matching index $j' \geq 1$, *such that* $r_{i',j'} = sync_{i'}(I)$, *and* $t_{\alpha_{i,j}} = t_{\alpha_{i',j'}}$. *Furthermore, there exists at least one* $i' \in I$, *such that* $t_{s_{i',j'}} = t_{\alpha_{i',j'}}$, i.e., *such that* $\Delta_{s_{i',j'}} = 0$, *for the matching index* j'.

Problem Formulation: Motion and Task Specifications

- Word and time sequences

Definition 3 (Words and time sequences)**.** *We denote by* $\upsilon_\tau = \varpi_1\varpi_2 \ldots = \mathcal{L}_i(\alpha_1)\mathcal{L}_i(\alpha_2)\ldots \in (2^{\Sigma_i} \cup 2^{\mathcal{E}_i})^\omega$ *the* service set sequence *associated with* τ*. The* word w_τ *produced by* τ *is the subsequence of the* non-silent *elements of* υ_τ*;* $w_\tau = \varpi_{\iota_1}\varpi_{\iota_2}\ldots \in (2^{\Sigma_i})^\omega$*, such that* $\varpi_1, \ldots, \varpi_{\iota_1 - 1} = \mathcal{E}_i$*, and for all* $j \geq 1$*,* $\varpi_{\iota_j} \neq \mathcal{E}_i$ *and* $\varpi_{\iota_j + 1}, \ldots, \varpi_{\iota_{j+1} - 1} = \mathcal{E}_i$*. With a slight abuse of notation, we use* $\mathbb{T}(\upsilon_\tau) = t_1t_2\ldots = t_{\alpha_1}t_{\alpha_2}\ldots$ *to denote the* service time sequence*, i.e. the subsequence of* $\mathbb{T}$ *when the services are provided. Furthermore,* $\mathbb{T}(w_\tau) = t_{\iota_1}t_{\iota_2}\ldots$ *denotes the* word time sequence*, i.e. the subsequence of* $\mathbb{T}(\upsilon_\tau)$ *that corresponds to the time instances when the non-silent services are provided.*

Problem Formulation: Motion and Task Specifications

- Service set at a time

Definition 4 (Service set at a time). *Let τ be a trace of $\mathcal{T}_i$ with $\upsilon_\tau = \varpi_1\varpi_2\ldots$, and $\mathbb{T}(\upsilon_\tau) = t_1t_2\ldots$. Given $t \in \mathbb{R}_0^+$, the service set $\upsilon_\tau(t) \in 2^{\Pi_i} \cup 2^{\mathcal{E}_i}$ provided at time t is $\upsilon_\tau(t) = \varpi_j$ if $t = t_j$ for some $j \geq 1$; and $\upsilon_\tau(t) = \mathcal{E}_i$ otherwise.*

Problem Formulation: Motion and Task Specifications

- Local LTL satisfaction

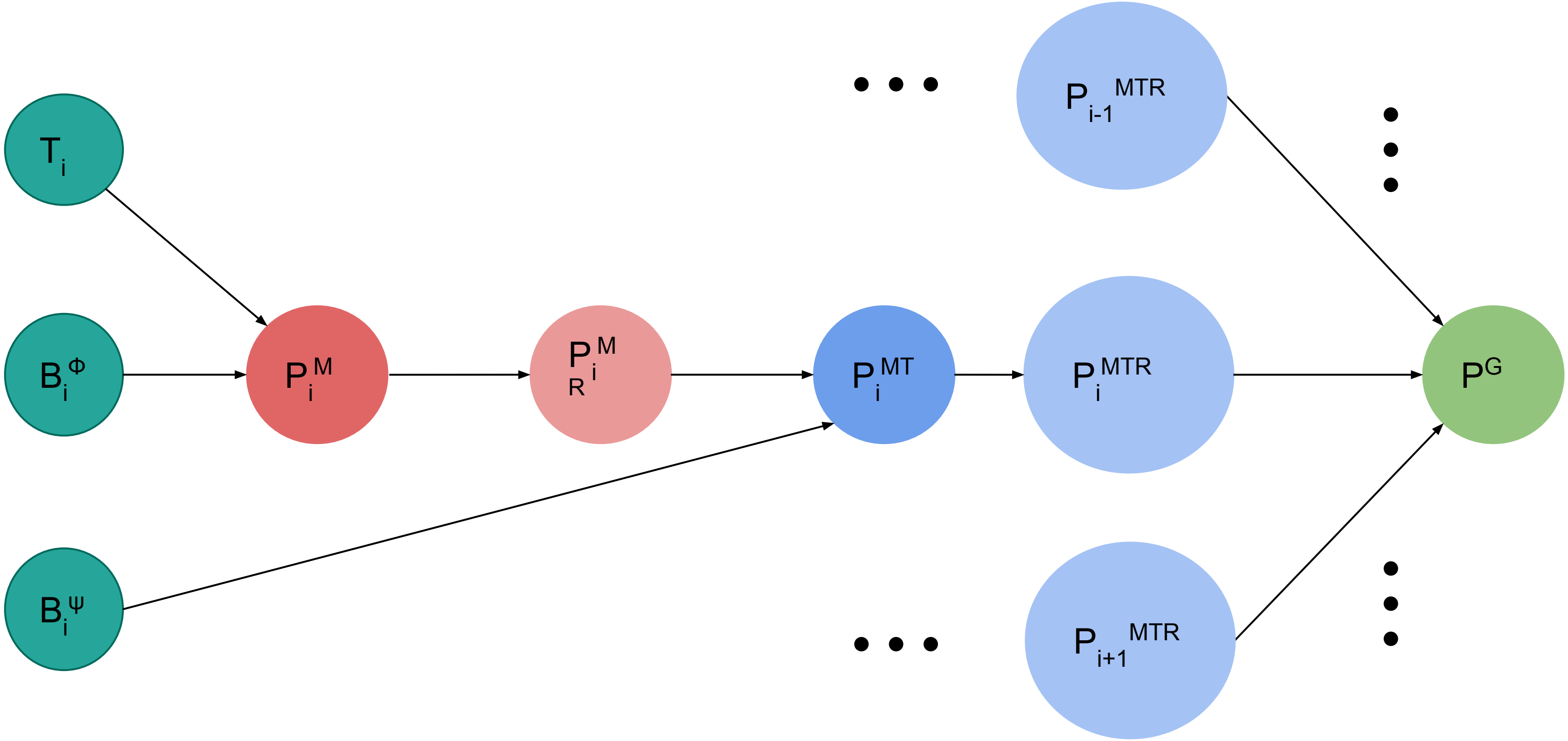
Definition 5 (Local LTL satisfaction). *Let* $\mathfrak{B} = \beta_1, \ldots, \beta_N$ *be a collection of behaviors, where* $\beta_i = (\tau_i, \gamma_i, \mathbb{T}_i)$ *for all* $i \in \mathcal{N}$ *and let* $\mathbb{T}(w_{\tau_i}) = t_{i,\iota_1} t_{i,\iota_2} \ldots$ *be the word time sequence of agent* i*. The* local word *produced by* $\mathfrak{B}$ *is* $w_{\mathfrak{B}_i} = \omega_{i,\iota_1}\omega_{i,\iota_2}\ldots$, *where* $\omega_{i,\iota_j} = \left(\bigcup_{i' \in \mathcal{N}} \upsilon_{\tau_{i'}}(t_{i,\iota_j})\right) \cap \Sigma$, *for all* $j \geq 1$. $\mathfrak{B}$ locally satisfies *the formula* ψ_i *for the agent* i, $\mathfrak{B} \models_i \psi_i$, *iff* $w_{\mathfrak{B}_i} \models \psi_i$.

Problem Formulation: Problem Statement

- Problem Statement

Problem 1. Consider *a set of agents* $\mathcal{N} = \{1, \ldots, N\}$, *and suppose that each agent* $i \in \mathcal{N}$ *is modeled as a tuple* $\mathcal{M}_i = (\mathcal{T}_i = (S_i, s_{init,i}, A_i, T_i, \Pi_i, L_i), Sync_i, \Sigma_i, \mathcal{L}_i)$, *and assigned a task in the form of an* $LTL_{\setminus X}$ *formula* ϕ_i *over* Π_i *and* ψ_i *over* $\mathbf{\Sigma} = \bigcup_{i' \in \mathcal{N}} \Sigma_{i'}$. For each $i \in \mathcal{N}$ find a strategy, i.e., *(i) a trace* $\tau_i = s_{i,1}\alpha_{i,1}s_{i,2}\alpha_{i,2}\ldots$ *of* $\mathcal{T}_i$ *and (ii) a synchronization sequence* γ_i *over* $Sync_i$ *with the property that the set of induced behaviors* $\mathbb{B}$ *from Def. 2 is nonempty, and for all* $\mathfrak{B} \in \mathbb{B}$ *and all* $i \in \mathcal{N}$, *it holds that the trace* τ_i *satisfies* ϕ_i*and the word* $w_{\mathfrak{B}_i}$ *produced by* $\mathfrak{B}$ *locally satisfies* ψ_i *for the agent* i *in terms of Def. 5.*

Problem Solution: Overview



Problem Solution: Preprocessing motion specifications

- Motion Product

Definition 6 (Motion product). *The* motion product *of a TS* $\mathcal{T}_i = (S_i, s_{init,i}, A_i, T_i, \Pi_i, L_i)$, *and a BA* $\mathcal{B}_i^\phi = (Q_i^\phi, q_{init,i}^\phi, \delta_i^\phi, 2^{\Pi_i}, F_i^\phi)$ *is a BA* $\mathcal{P}_i = (Q_i, q_{init,i}, \delta_i, 2^{\Sigma_i} \cup \{\mathcal{E}_i\}, F_i)$, *where* $Q_i = S_i \times Q_i^\phi$; $q_{init,i} = (s_{init,i}, q_{init,i}^\phi)$; $((s,q), \mathcal{L}_i(\alpha), (s', q')) \in \delta_i$ *if and only if* $(s, \alpha, s') \in T_i$, *and* $(q, L_i(s), q') \in \delta_i^\phi$; *and* $F_i = \{(s,q) \mid q \in F_i^\phi\}$.

- Insignificant States

Definition 7 (Insignificant states in $\mathcal{P}_i$). *A state p of the motion product $\mathcal{P}_i$ is* significant *if it is (i) the initial state $p = q_{init,i}$, or (ii) if there exists a transition $(p, \sigma, p') \in \delta_i$, such that $\sigma \neq \mathcal{E}_i$; and* insignificant *otherwise.*

Problem Solution: Preprocessing motion specifications

Algorithm 1 Reduction of the motion product

Input: motion product $\mathcal{P}_i = (Q_i, q_{init,i}, \delta_i, 2^{\Sigma_i} \cup 2^{\mathcal{E}_i}, F_i)$
Output: reduced BA $\ddot{\mathcal{P}}_i = (\ddot{Q}_i, \ddot{q}_{init,i}, \ddot{\delta}_i, 2^{\Sigma_i} \cup 2^{\mathcal{E}_i}, \ddot{F}_i)$

1: initialize $\ddot{\mathcal{P}}_i := \mathcal{P}_i$
2: **for all** insignificant states $p \in Q_i \setminus F_i$ **do**
3: $\quad \ddot{Q}_i := \ddot{Q}_i \setminus \{p\}$
4: $\quad \ddot{\delta}_i := \ddot{\delta}_i \setminus \{(p', \sigma, p), (p, \sigma, p') | p' \in \ddot{Q}_i, \sigma \in 2^{\Sigma_i} \cup 2^{\mathcal{E}_i}\}$
5: $\quad \ddot{\delta}_i := \ddot{\delta}_i \cup \{(p', \sigma, p'') \mid p', p'' \in \ddot{Q}_i, \sigma \in 2^{\Sigma_i} \cup 2^{\mathcal{E}_i},$ $(p', \sigma, p), (p, \mathcal{E}_i, p'') \in \ddot{\delta}_i\}$
6: **end for**
7: **for all** insignificant states $p \in F_i$, such that all predecessors of p in $\ddot{\mathcal{P}}_i$ are insignificant **do**
8: $\quad$ **if** $(p, \mathcal{E}_i, p) \in \ddot{\delta}_i$ **then**
9: $\quad\quad \ddot{\delta}_i := \ddot{\delta}_i \cup \{(p', \mathcal{E}_i, p') \mid (p', \mathcal{E}_i, p) \in \ddot{\delta}_i\}$
10: $\quad$ **end if**
11: $\quad \ddot{Q}_i := \ddot{Q}_i \setminus \{p\}$
12: $\quad \ddot{\delta}_i := \ddot{\delta}_i \setminus \{(p', \sigma, p), (p, \sigma, p') | p' \in \ddot{Q}_i, \sigma \in 2^{\Sigma_i} \cup 2^{\mathcal{E}_i}\}$
13: $\quad \ddot{\delta}_i := \ddot{\delta}_i \cup \{(p', \mathcal{E}_i, p'') \mid p', p'' \in \ddot{Q}_i, (p', \mathcal{E}_i, p), (p, \mathcal{E}_i, p'')$ $\in \ddot{\delta}_i\}$
14: **end for**

Problem Solution: Preprocessing task specifications

- Task and motion product

Definition 8 (Task and motion product). *The* task and motion product *of a reduced motion product automaton* $\ddot{\mathcal{P}}_i = (\ddot{Q}_i, \ddot{q}_{init,i}, \ddot{\delta}_i, 2^{\Sigma_i} \cup 2^{\mathcal{E}_i}, \ddot{F}_i)$*, and the task specification BA* $\mathcal{B}_i^{\psi} = (Q_i^{\psi}, q_{init,i}^{\psi}, \delta_i^{\psi}, 2^{\mathbf{\Sigma}}, F_i^{\psi})$ *is a BA* $\bar{\mathcal{P}}_i = (\bar{Q}_i, \bar{q}_{init,i}, \bar{\delta}_i, 2^{\mathbf{\Sigma}} \cup 2^{\mathcal{E}_i}, \bar{F}_i)$*, where* $\bar{Q}_i = \ddot{Q}_i \times Q_i^{\psi} \times \{1, 2, 3\}$; $\bar{q}_{init,i} = (q_{init,i}, q_{init,i}^{\psi}, 1)$; $\bar{F}_i = \{(q_1, q_2, 2) \mid q_2 \in F_i^{\psi}\}$; *and* $((q_1, q_2, j), \sigma, (q_1', q_2', j') \in \bar{\delta}_i$ *iff*

- $\sigma = \mathcal{E}_i$, $(q_1, \mathcal{E}_i, q_1') \in \ddot{\delta}_i$, *and* $q_2 = q_2'$*; or*
- $\sigma \in 2^{\mathbf{\Sigma}}$, $(q_1, \sigma \cap 2^{\Sigma_i}, q_1') \in \ddot{\delta}_i$*, and* $(q_2, \sigma, q_2') \in \delta_i^{\psi}$,

and $j' = 2$*, if* $j = 1$ *and* $q_1' \in \ddot{F}_i$, $j' = 3$ *if* $j = 2$ *and* $q_2' \in \ddot{F}_i^{\psi}$, $j' = 1$ *if* $j = 3$*, and* $j = j'$ *otherwise.*

Problem Solution: Preprocessing task specifications

- Assisting services

Definition 9 (Assisting services). *Suppose that* $i \neq i'$. *A service* $\rho \in \Sigma_{i'}$ *is* not assisting on a transition $(p, \sigma, p') \in \bar{\delta}_i$ *of* $\bar{\mathcal{P}}_i$ *if and only if it holds that* $(p, \sigma \cup \{\rho\}, p') \in \bar{\delta}_i \iff (p, \sigma \setminus \{\rho\}, p') \in \bar{\delta}_i$; *it is* assisting on (p, σ, p') *otherwise.* $Dep_i(\mathfrak{t}) = \{i\} \cup \{i' \mid i' \neq i \text{ and } \exists \rho \in \Sigma_{i'} \text{ assisting on } \mathfrak{t}\}$.

- Globally assisting services

Definition 10 (Globally assisting services). *A service* $\rho \in \Sigma_{i'}$ *of agent* $i' \in \mathcal{N}$ *is* globally assisting *if there exists* $i \in \mathcal{N}$, $i \neq i'$, *and a transition* $\mathfrak{t} \in \bar{\delta}_i$, *such that* ρ *is assisting on* $\mathfrak{t}$.

Problem Solution: Preprocessing task specifications

- Insignificant states

Definition 11 (Insignificant states in $\bar{\mathcal{P}}_i$). *A state p of the task and motion product $\bar{\mathcal{P}}_i$ is* significant *if it is either (i) the initial state $p = \bar{q}_{init,i}$, or (ii) if there exists a transition $(p, \sigma, p') \in \bar{\delta}_i$, such that there exists a globally assisting service $\rho \in \sigma \cap \Sigma_i$, or (iii) if there exists a transition $(p, \sigma, p') \in \bar{\delta}_i$, such that $Dep_i(p, \sigma, p') \neq \{i\}$; the state p is* insignificant *otherwise.*

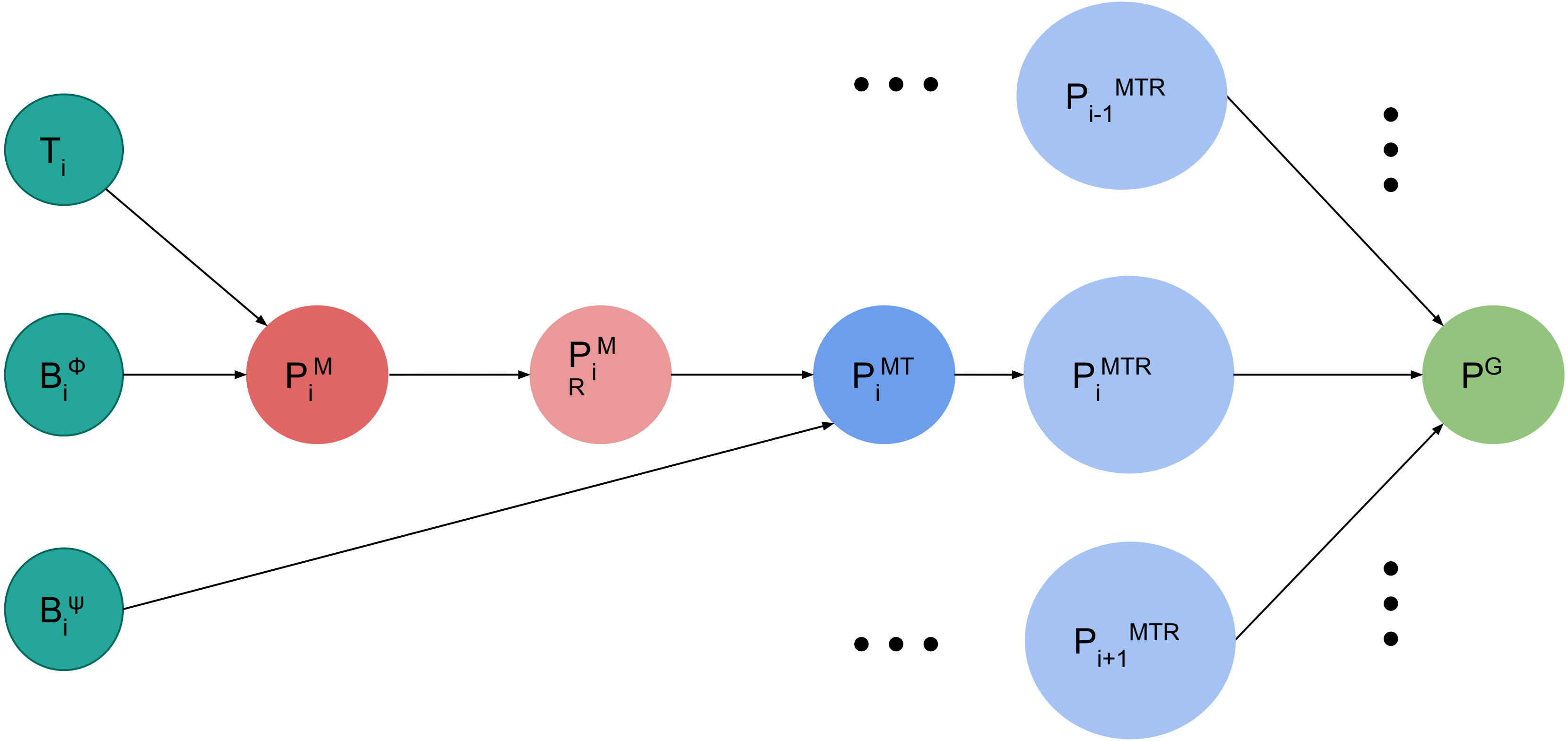
Problem Solution: Preprocessing task specifications

- Global Product

Definition 12 (Global product). *The* global product *of the reduced task and motion product automata* $\widehat{\mathcal{P}}_1, \ldots \widehat{\mathcal{P}}_N$, *where* $\widehat{\mathcal{P}}_i = (\widehat{Q}_i, \widehat{q}_{init,i}, \widehat{\delta}_i, 2^{\mathbf{\Sigma}} \cup 2^{\mathcal{E}_i}, \widehat{F}_i)$, *for all* $i \in \mathcal{N}$, *is a BA* $\mathcal{P} = (Q, q_{init}, \delta, 2^{\mathbf{\Sigma}} \cup 2^{\mathcal{E}=\{\mathcal{E}_i | i \in \mathcal{N}\}}, F)$ *with a mapping* $Dep : \delta \to 2^{\mathcal{N}}$, *where* $Q = \widehat{Q}_1 \times \ldots \times \widehat{Q}_N \times \{1, \ldots, N+1\}$; $q_{init} = (\widehat{q}_{init,1}, \ldots, \widehat{q}_{init,N}, 1)$; $\bar{F}_i = \{(q_1, \ldots, q_N, N) \mid q_N \in \widehat{F}_N\}$; *and* $\mathsf{t} = ((q_1, \ldots, q_N, j), \sigma, (q'_1, \ldots, q'_N, j')) \in \delta$ *iff either*

- $\exists i \in \mathcal{N}$, *such that* $\sigma = \mathcal{E}_i$, $(q_i, \mathcal{E}_i, q'_i) \in \widehat{\delta}_i$, $q_{i'} = q'_{i'}$, *for all* $i' \neq i$, *and* $j' = j + 1$ *if* $j = i$ *and* $q'_i \in \widehat{F}_i$, $j' = 1$ *if* $j = N + 1$, *and* $j' = j$ *otherwise. Then we set* $Dep(\mathsf{t}) = \{i\}$; *or*
- $\sigma \in 2^{\mathbf{\Sigma}}$, *and* $\exists \mathcal{I} \subseteq \mathcal{N}$, *such that for all* $i \in \mathcal{I}$ *it holds that* $(q_i, \sigma, q'_i) \in \widehat{\delta}_i$ *while for all* $i \notin \mathcal{I}$ *it holds that* $q_i = q'_i$. *Moreover,* $\bigcup_{i \in \mathcal{I}} Dep_i(q_i, \sigma, q'_i) \subseteq \mathcal{I}$, *and* $j' = j + 1$ *if* $j \in \mathcal{I}$ *and* $q'_j \in \widehat{F}_j$, $j' = 1$ *if* $j = N + 1$, *and* $j' = j$ *otherwise. Then we set* $Dep(\mathsf{t}) = \mathcal{I}$.

Problem Solution: Overview



Problem Formulation: Problem Statement

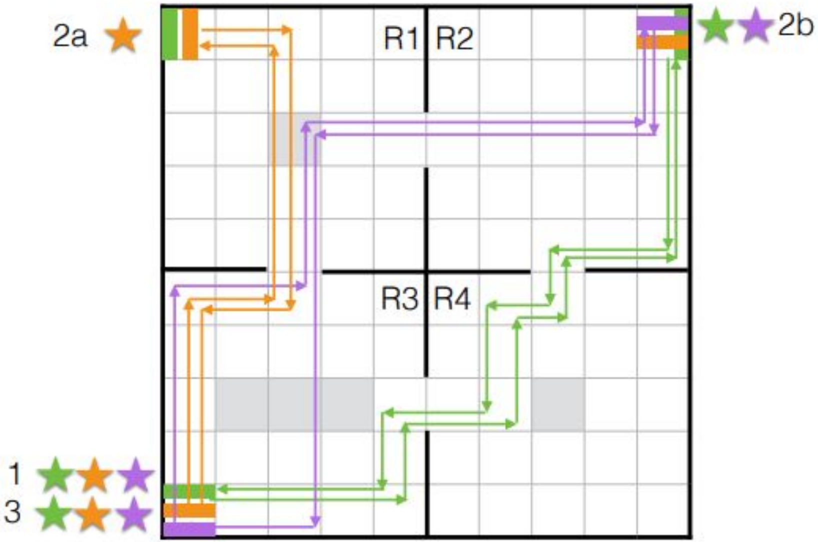
- Problem Statement

Problem 1. Consider *a set of agents* $\mathcal{N} = \{1, \ldots, N\}$, *and suppose that each agent* $i \in \mathcal{N}$ *is modeled as a tuple* $\mathcal{M}_i = (\mathcal{T}_i = (S_i, s_{init,i}, A_i, T_i, \Pi_i, L_i), Sync_i, \Sigma_i, \mathcal{L}_i)$, *and assigned a task in the form of an* $LTL_{\backslash X}$ *formula* ϕ_i *over* Π_i *and* ψ_i *over* $\mathbf{\Sigma} = \bigcup_{i' \in \mathcal{N}} \Sigma_{i'}$. For each $i \in \mathcal{N}$ find a strategy, i.e., *(i) a trace* $\tau_i = s_{i,1}\alpha_{i,1}s_{i,2}\alpha_{i,2} \ldots$ *of* $\mathcal{T}_i$ *and (ii) a synchronization sequence* γ_i *over* $Sync_i$ *with the property that the set of induced behaviors* $\mathbb{B}$ *from Def. 2 is nonempty, and for all* $\mathfrak{B} \in \mathbb{B}$ *and all* $i \in \mathcal{N}$, *it holds that the trace* τ_i *satisfies* ϕ_i*and the word* $w_{\mathfrak{B}_i}$ *produced by* $\mathfrak{B}$ *locally satisfies* ψ_i *for the agent* i *in terms of Def. 5.*

Problem Solution: Strategy Synthesis

1. Generate accepting run for the global product automata
2. Map the states of global product to the accepting run of reduced motion-task product automata and corresponding dependency set for synchronization
3. Map the accepting run of reduced motion-task product automata to the accepting run of motion-task product automata with singleton *sync* sets
4. Map the accepting run of motion-task product automata to reduced motion product automata
5. Map the accepting run of reduced motion-task product automata to motion product automata
6. Map the accepting run of motion product automata to trace of TS_i

Cooperative Planning Example

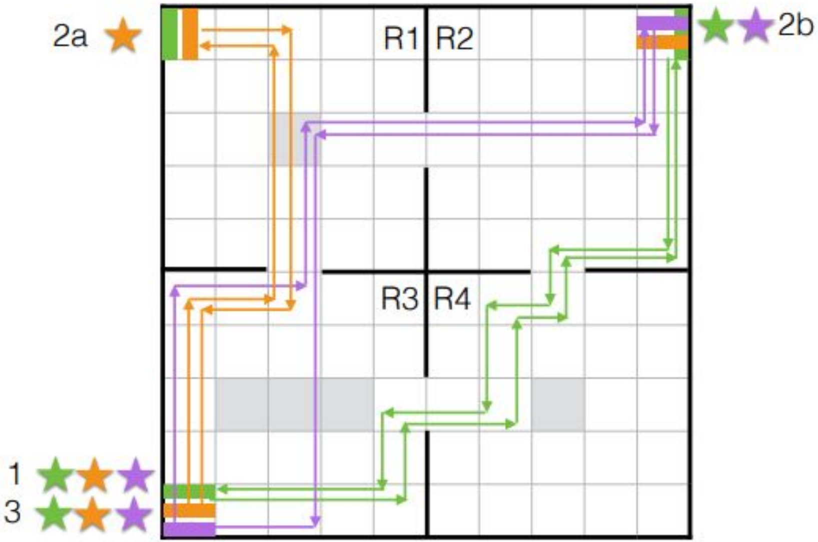


Agent 1 (Green) :
Land Vehicle
Services: *load* and *unload*

Agent 2 (Orange) :
UAV
Services: *help* and *inform*

Agent 3 (Purple) :
UAV
Services: *assist*

Cooperative Planning Example: Motion Specifications



Agent 1 (Green) :

$\Phi_1 = \mathrm{G}\neg \mathrm{R1}$

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

Agent 2 (Orange) :

$\Phi_2 = \mathrm{G}\neg \mathrm{R2}$

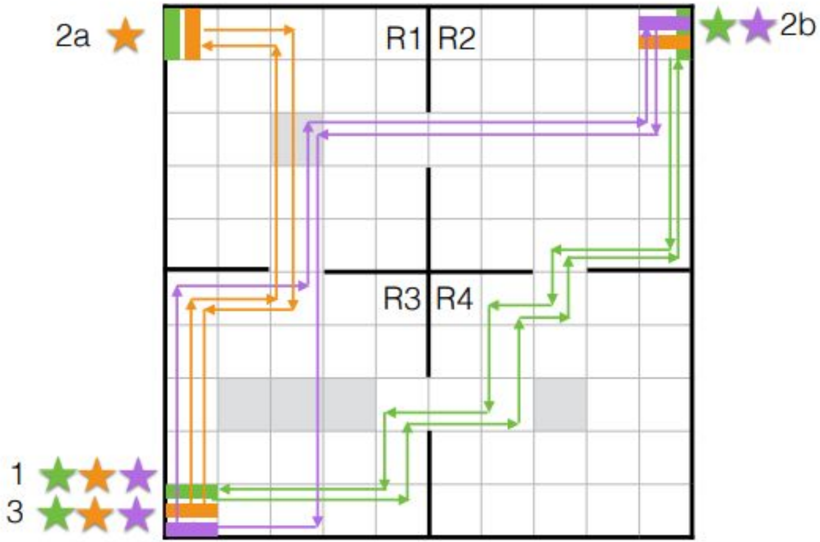
$\psi_2 = \mathrm{GF}\ inform$

Agent 3 (Purple) :

$\Phi_3 = \mathrm{GF\ R1} \wedge \mathrm{GF\ R1}$

$\psi_3 = assist \vee \neg assist$

Cooperative Planning Example: Motion Specifications



Centralized Planning:

Synchronous product of TSs ~ 100,000

Distributed Planning:

BAs for given formulas have 2,2,3,2,2,1 states
Intersection BA states ~ 330 states
Overall product automaton ~ 33 million

Reduced PAs have 27, 17 and 8 states
Global PA has 15000 states

Extensions

Extensions

Cooperative Decentralized Multi-agent Control Under LTL Task Under Connectivity Constraints

We will find motion controllers and action plans for the agents that guarantee the satisfaction of all individuals LTL tasks.

While designing the controllers two things into consideration has been taken :-

Extensions

1. Communication Constraints

$$\| x_i(t) - x_{\square}(t) \| \leq r$$

Or there exist k s.t

$$\| x_i(t) - x_{\square}(t) \| \leq r \quad \text{where k and j are connected}$$

(inter communication)

Extensions

2. Leader Selection

Goal is to select leader and all other agents will follow it as long as it’s task is completed and after completion of its task the new leader will be selected.

2.1 offline step

2.2 online step

2.3 urge function

Extensions

- Hybrid Control of Multi-Agent Systems
- Main contributions of the proposed hybrid control strategy lie in two aspects:
  - the proposed distributed motion controller guarantees almost global convergence and the satisfaction of relative-distance constraints at all time, for an arbitrary number of leaders with different local goals in the team.
  - two different local coordination and control-law switching policies are proposed depending on the types of local tasks assigned.

Extensions

The proposed solution consists of three layers :

- An offline synthesis algorithm for discrete plan of each agent.
- A continuous control scheme that guarantees one of the agent reaches its progressive goal region in finite time while relative-distance constraints are fulfilled.
- A hybrid control layer that coordinates the discrete plan execution and continues control law switching in running time.

Extensions

- Discrete Plan Synthesis
  - The discrete plan can be generated using standard techniques leveraging ideas from automata based formal verification.
  - Current temporal logic based discrete plan synthesis can accommodate various environmental constraints and advanced plan optimality criteria .
- Continuous Controller Design
  - we use notion of Connectivity graph which allows us to handle the relative-distance constraints.
  - we use a Relative-Distance maintenance which uses potential field function for an attractive potential to agent i's neighbours.

Extensions

- Hybrid Controller Structure
  - Here, we can extend to two different switching protocols for each agent to decide its own activity or passivity under different cases, such that all agents can fulfill their local tasks and at the same time satisfy relative distance constraints.

  - This can be demonstrated through computer simulations.

Extensions

- Cooperative Task Planning of Multi-Agent Systems Under Timed Temporal Specifications:
    - Treating Time in quantitative manner
    - Using Metric Interval Temporal Logic
    - Automatic control synthesis in two-stage systematic procedure

Extensions

- Multi-robot LTL Planning Under Uncertainty:
  - Modelling the partially known environments
  - Decentralized Planner over global and local missions
  - Breaks down the the set of robots into classes based on specification dependencies
  - Makes two calls to the classical planner
  - Searches for definitive and possible plans
  - Chooses between definitive and possible plans during execution