Coordination of temporal plans in dynamic environments for mobile agents

Amal El Fallah Seghrouchni
Amal.Elfallah@lip6.fr
Presentation outline

• Introduction
  • Multi-Agent Context
  • (Multi-Agent) Temporal planning
  • Multi-Agent Planning & Coordination

• Framework 1: multi-agent temporal planning and coordination based on Coordinated-SAPA
  • Coordination during planning
  • Coordination after planning
  • P-CLAIM

• Framework 2: overview of multi-agent temporal planning and coordination based on Hybrid Automata
  • Tactical aircraft simulation
  • Our 5-phases approach

• Conclusion and perspectives
Multi-agent context

- Agents are mobile and operate in a shared and dynamic environment
  - e.g. planes, UAVs, UCAV, robots, ants, etc.
- Agents are involved in a mission (goals to be achieved)
  - individual goals and /or shared global goal(s)
- Agents are lead to compute plan(s)
  - individual and/or multi-agent plans
  - coordination with other agents 'plans is necessary
- Time is crucial for agents
(Multi-Agent) Temporal planning

- Temporal Planning: take into account the duration of actions while computing a plan
  - Position constrained plans: Start time of each action ([Do 2001], [Bacchus 2001], [Smith 1999], [Edelkamp 2001])
  - Order constrained plans: Precedence constraints between actions ([Penberthy 1995], [Muscettola 1993], [Labone 1995])

- Multi-Agent Temporal Planning
  - Reasoning about other agents (durations of) actions
  - Synchronization with agents’ plans and actions
  - Reasoning about possible interferences
Coordination by means of Multi-Agent Planning

- **Centralised Planning**
  - 1 Planner
  - Several agents execute plan(s)

- **Distributed Planning**
  - Each agent plans & executes

**Framework 1**
- Task oriented
  - Decomposition
- Agent oriented
  - Coordination

**Framework 2**

Examples

Task oriented coordination

Agent oriented coordination

Coordination
Coordination by means of Multi-Agent Planning

- Removing conflicts (negative interactions)
- Utilising help relations (positive interactions)

The question is: **When the plans should be coordinated?**
- Before, during or after planning?
  - *Time is still crucial!*
  - *The environment is shared and dynamic!*
  - *Plans are concurrent!*
Related Work (Plans Coordination)

- **Coordination before planning**
  - Social Laws [Shoham 1995], [Buzzing 2006]

- **Coordination during planning**
  - (Global) Partial Global Planning [Durfee and Lesser 1987]
  - Incremental Plan Merging [Alami et al. 1994]
  - Recursive Petri Nets [El Fallah Seghrouchni and Haddad 1996]

- **Coordination after planning**
  - Temporal Plan Merging [Tsamardinos et al. 2000]
  - Merging Hierarchical Plans [von Martial 1992]
  - Synchronization of a network of hybrid automata [El Fallah Seghrouchni et al., 2004]
Presentation outline

- **Introduction**
  - Multi-Agent Context
  - (Multi-Agent) Temporal planning
  - Multi-Agent Planning & Coordination

- **Framework 1: multi-agent temporal planning and coordination based on Coordinated-SAPA**
  - Coordination during planning
  - Coordination after planning
  - P-CLAIM

- **Framework 2: overview of multi-agent temporal planning and coordination based on Hybrid Automata**
  - Tactical aircraft simulation
  - Our 5-phases approach

- **Conclusion and perspectives**
Framework #1

Temporal Planning and the Plan Coordination Mechanisms

Ph.D. Thesis of Muhammad Adnan HASHMI

(24-01-2012)
Objectives (Plans Coordination)

- Propose plans coordination mechanisms for the plans having different priorities

- Two different scenarios
  - Coordination during planning
    - Coordinated Planning Problem (CPP)
  - Coordination after planning
    - Proactive-Reactive Coordination Problem (PRCP)
Assumptions

- Two agents $\alpha$ and $\beta$ sharing the same environment
  - Agent $\alpha$ having higher priority (reactive) goals
  - Agent $\beta$ having normal priority (proactive) goals

- Two possible conflicts among plans
  - Causal link threat
  - Parallel actions interference
Two Possible Conflicts

- **Causal Link** \((A_1, A_2, p)\)
  - Action \(A_1\) adds an effect \(p\)
  - Action \(A_2\) needs this effect
  - No action between \(A_1\) and \(A_2\) adding \(p\)

- **Causal Link Threat**
  - If an action \(A\) deletes \(p\) and lies between \(A_1\) and \(A_2\), then \(A\) threatens the causal link \((A_1, A_2, p)\)
Two Possible Conflicts

- **Causal Link** \((A_1, A_2, p)\)
  - Action \(A_1\) adds an effect \(p\)
  - Action \(A_2\) needs this effect
  - No action between \(A_1\) and \(A_2\) adding \(p\)

- **Parallel Actions Interference**
  - Actions \(A_1\) and \(A_2\) lie in parallel
  - Either one of them deletes the preconditions or add effects of the other

\[\text{Diagram: Causal Link and Parallel Actions Interference}\]
Coordinated Planning Problem (CPP)

Scenario 1: coordination during planning

- **Prerequisite:**
  - Plan $P_\alpha$ of Agent $\alpha$

- **Our Aim:**
  - Compute a Plan $P_\beta$ for Agent $\beta$
    - Has no conflict with $P_\alpha$
    - Avails the cooperative opportunities offered by $P_\alpha$

- **Solution: Coordinated-Sapa**
  - Extension of the well-known temporal planner *Sapa* [M. Do & S. Kambhampati 2001]
Sapa: Multi-Objective Heuristic Based Temporal Planner [Do and Kambhampati 2001]

A* Search through the space of time-stamped states

\[ S = (P, M, \Pi, Q, t) \]

Set \( <p_i, t_i> \) of predicates \( p_i \) and the time of their last achievement \( t_i < t \).

Set of protected persistent conditions.

Time stamp of \( S \).

Event queue containing delayed effects.

Set of functions represent resource values.
Main Flow-Diagram of SAPA

\[ S = (P, M, \Pi, Q, t) \]

\[ S = (P = I, M, \Pi, Q = \varnothing, t = 0) \]

G = (\langle p_1; t_1 \rangle, \ldots, \langle p_n; t_n \rangle)

Generate start state

Queue of Time-Stamped states

Select state with lowest H-value

Expand state by applying actions

Build RTPG: Relaxed Temporal Planning Graph

Propagate Cost functions
Extract relaxed plan
Adjust for Mutexes; Resources

Satisfies Goals?

Print Plan

Heuristic estimation
Some Important Concepts

\[ S = (P, M, \Pi, Q, t) \]

- **Goal Satisfaction**

  \[ S = (P, M, \Pi, Q, t) \Rightarrow G \text{ if } \forall <p_i, t_i> \in G \text{ either:} \]
  
  - \( \exists <p_i, t_j> \in P, t_j < t_i \) and no event in \( Q \) deletes \( p_i \)
  
  - \( \exists e \in Q \) that adds \( p_i \) at time \( t_e < t_i \)

- **Action Application**

  Action \( A \) is applicable in \( S = (P, M, \Pi, Q, t) \) if:
  
  - All preconditions of \( A \) are satisfied by \( P \)
  
  - \( A \)’s effects do not interfere with \( Q \)

  When \( A \) is applied to \( S = (P, M, \Pi, Q, t) \):
  
  - \( P \) is updated according to \( A \)’s instantaneous effects
  
  - Delayed effects of \( A \) are put in \( Q \).

- **Special Advance-Time Action**

  - Advances time to next earliest event in the queue, and adds the event to \( P \)
Coordinated-Sapa

[Hashmi M.A. & El Fallah Seghrouchni A., 2010a]

An Extension of Sapa that Finds Coordinated Plans in Temporal Domains
Coordinated-\textit{Sapa}

- Our extension of the well-known temporal planner \textit{Sapa}
- Input
  - Initial State $I$
  - A Set of Actions $D$
  - A Set of Proactive Goals $G_P$ with deadlines
  - A Reactive Plan $P_R$
- Output
  - A plan $P_P$ from $I$ to $G_P$
    - $P_P$ is consistent with $P_R$
    OR
  - Deadline Violated $\Rightarrow$ Failure + Goal name whose deadline is violated
    OR
  - No Solution Possible $\Rightarrow$ Failure
Handling Positive Interactions

• Before starting planning for Agent $\beta$
  
  • Create an event $e = (st, et, effect)$ corresponding to every effect $k$
    of every action $A$ in $P_\alpha$
    
    • $e.st = st(A)$
    • $e.et = et(A)$
    • $e.effect = k$

  • Put all these events in the event queue $Q$

• WHY? At every state, Coordinated-Sapa will take into
  account all the changes made by Agent $\alpha$

  • Advance-Time Action will add the effects generated by Agent $\alpha$, to $P$
Handling Negative Interactions

\[ S = (P, M, \Pi, Q, t) \]

- Add another action applicability condition to the planning mechanism of Agent \( \beta \)

- **Action Application:**
  - Action \( A \) is applicable in \( S = (P, M, \Pi, Q, t) \) if:
    - All preconditions of \( A \) are satisfied by \( P \)
    - \( A \)'s effects do not interfere with \( Q \)
    - \( A \) does not threaten any causal link (of \( P_\alpha \)) at \( t \)
Example

- System:
  - 1 car Car1 and 1 plane Plane1; 4 persons P1, P2, P3 and P4.

- Goals
  - agent Car1: carry P4 to its destination,
  - agent Plane1: carry P1, P2 and P3 to their destinations.

- The initial and goal states for agent Plane1 are as follows:
  - I = at(P4;C4); at(Plane1;C3); at(Car1;C4)
  - G = at(P4;C5)

- The initial and goal states for agent Car1 are as follows:
  - I = at(P1;C1); at(P2;C1); at(P3;C2); at(Plane1;C3); at(Car1;C4)
  - G = at(P1;C2); at(P2;C2); at(P3;C5)
A pair plans Generated by Coordinated-SAPA

<table>
<thead>
<tr>
<th></th>
<th>Fly(P1 C3 C1)</th>
<th>Board(P1 Pl1)</th>
<th>Fly(P1 C1 C2)</th>
<th>Disembark(P1 Pl1)</th>
<th>Fly(P1 C2 C5)</th>
<th>Disembark(P3 Pl1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Board(P2 Pl1)</td>
<td></td>
<td>Disembark(P2 Pl1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Board(P3 Pl1)</td>
</tr>
<tr>
<td>t=0</td>
<td>180</td>
<td>200</td>
<td>250</td>
<td>380</td>
<td>390</td>
<td>400</td>
</tr>
<tr>
<td>580</td>
<td>590</td>
<td>600</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Agent β** drives person **P4** in **Car1** to city **C2**
  - So that **P4** can board the plane **Pl1**
  - Agent **α** is bringing the plane **Pl1** to **C2** from **C3**

- **Agent β** makes person **P4** board the plane **Pl1** at **C2**
  - Agent **α** flies plane **Pl1** from **C2** to **C5**
Experimental Results

- Domains and problems taken from 3rd International Planning Competition
- A multi-agent problem is generated by taking the original problem and dividing the goals in two sets, one for each agent

<table>
<thead>
<tr>
<th>No</th>
<th>Problem</th>
<th>Number of Goals α/β</th>
<th>Actions Pair of Plan</th>
<th>Makespan Pair of Plan</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Zeno4</td>
<td>3/2</td>
<td>(7,3)</td>
<td>(643,330)</td>
</tr>
<tr>
<td>2</td>
<td>Zeno5</td>
<td>2/2</td>
<td>(8,8)</td>
<td>(660,1216)</td>
</tr>
<tr>
<td>3</td>
<td>Zeno6</td>
<td>2/3</td>
<td>(7,6)</td>
<td>(540,950)</td>
</tr>
<tr>
<td>4</td>
<td>Zeno7</td>
<td>4/2</td>
<td>(9,7)</td>
<td>(736,1146)</td>
</tr>
<tr>
<td>5</td>
<td>Zeno8</td>
<td>3/4</td>
<td>(4,7)</td>
<td>(330,950)</td>
</tr>
<tr>
<td>6</td>
<td>Zeno9</td>
<td>2/5</td>
<td>(8,17)</td>
<td>(693,1186)</td>
</tr>
<tr>
<td>7</td>
<td>Zeno10</td>
<td>3/3</td>
<td>(10,--)</td>
<td>(570,--)</td>
</tr>
<tr>
<td>8</td>
<td>Depot52</td>
<td>2/2</td>
<td>(9,7)</td>
<td>(23,57)</td>
</tr>
<tr>
<td>9</td>
<td>Depot53</td>
<td>3/3</td>
<td>(--)</td>
<td>(--)</td>
</tr>
<tr>
<td>10</td>
<td>DriverLog2</td>
<td>4/3</td>
<td>(12,6)</td>
<td>(88,99)</td>
</tr>
<tr>
<td>11</td>
<td>DriverLog4</td>
<td>5/4</td>
<td>(7,10)</td>
<td>(45,87)</td>
</tr>
<tr>
<td>12</td>
<td>DriverLog5</td>
<td>5/3</td>
<td>(11,9)</td>
<td>(66,78)</td>
</tr>
<tr>
<td>13</td>
<td>DriverLog6</td>
<td>5/5</td>
<td>(6,5)</td>
<td>(40,51)</td>
</tr>
<tr>
<td>14</td>
<td>DriverLog7</td>
<td>5/5</td>
<td>(7,6)</td>
<td>(36,37)</td>
</tr>
<tr>
<td>15</td>
<td>DriverLog8</td>
<td>7/4</td>
<td>(--)</td>
<td>(--)</td>
</tr>
<tr>
<td>16</td>
<td>DriverLog9</td>
<td>4/6</td>
<td>(16,--)</td>
<td>(38,--)</td>
</tr>
<tr>
<td>17</td>
<td>DriverLog10</td>
<td>4/4</td>
<td>(13,8)</td>
<td>(37,51)</td>
</tr>
<tr>
<td>18</td>
<td>Rover2</td>
<td>2/1</td>
<td>(5,3)</td>
<td>(29,44)</td>
</tr>
<tr>
<td>19</td>
<td>Rover4</td>
<td>2/1</td>
<td>(5,3)</td>
<td>(30,45)</td>
</tr>
<tr>
<td>20</td>
<td>Rover5</td>
<td>3/4</td>
<td>(9,11)</td>
<td>(43,98)</td>
</tr>
<tr>
<td>21</td>
<td>Rover6</td>
<td>5/5</td>
<td>(18,--)</td>
<td>(102,--)</td>
</tr>
</tbody>
</table>

*Coordinated-Sapa* performance on multi-agent problems.
Proactive-Reactive Coordination

Scenario 2: Coordination after planning

- Prerequisite:
  - Reactive plan $P_\alpha$ of Agent $\alpha$
  - Proactive plan $P_\beta$ of Agent $\beta$

- Our Aim:
  - Modify plan $P_\beta$ such that:
    - It has no conflict with $P_\alpha$
    - Avails the cooperative opportunities offered by $P_\alpha$

- Solution:
  - Plan Merging Algorithm
Case Study

- **Goals of Rescue Agent**
  - Rescue the victims
- **Goals of Analyzer Agent**
  - Analyze the goal cells
  - Call the central agent
- **Constraints**
  - One agent in a cell
  - Hyper energy cells: needs fuel or energy to enter
  - Agent should have key to open door
Conflict Resolution

- Threat-Repair Link \((A_1, A_2, p)\)
  - Action \(A_1\) deletes \(p\)
  - \(A_2\) is a subsequent action and adds \(p\)

- \(A_1\) is called Threat Action
- \(A_2\) is called Repair Action
Valid and Possibly Valid Time Stamps

- Possibly Valid Time Slot for an action $A$
  - All preconditions are met
  - No parallel actions interference

- Valid Time Slot for an action $A$
  - All preconditions are met
  - No parallel actions interference
  - Either:
    - No causal link threat
    - Repair Action exists before the deadline
Plan Merging Algorithm

- Fix all the actions of Reactive Plan \( P_\alpha \) on timeline

- For every action \( CA \) of Proactive Plan
  - Search for the first Possibly Valid Time Slot \( T \) on timeline
  - Reason about the time slot \( T \)
    - There could be 5 cases at \( T \)
Plan Merging Algorithm

Case 1: No causal link threat by CA at T
- Assign Time Slot T to CA

EXAMPLE
- Current Action: \texttt{Move(A1, A2)}
  - Returned Time Slot: 0 - 5
  - Any Threat? : \texttt{No}
  - Assign Time Slot 0 – 5 to CA
Plan Merging Algorithm

Case 2: CA threatens a Causal Link but Repair Action exists
- Assign Time Slot T to CA
- Save a Possible Threat <ThreatAction, RepairAction, Deadline>

EXAMPLE
- Current Action: Move(A4, A5)
  - Time Slot: 20 - 25
  - Any Threat? : Yes (Agent $\alpha$ needs A5 at time 40-45)
  - Repair Action: Move(A5, A6)
  - Assign Time Slot 20 - 25 to Move (A4, A5)
  - Save <Move(A4, A5), Move(A5, A6), 40>
Plan Merging Algorithm

Case 3: It is a Repair Action but can not meet a deadline of some Threat Action
  - Backtrack to the Threat Action, find another time stamp

EXAMPLE
  - Current Action: Move (A8, A9)
    - Returned Time Slot: 50 - 55
    - Any Threat?: Yes (Agent $\alpha$ needs A9 at 85-110)
    - Repair Action: Move (A9, B9)
    - Save <Move(A8, A9), Move(A9, B9), 85>
  - Next Action: AnalyzeCell (A9)
    - Time Slot Assigned: 55 - 70
Plan Merging Algorithm

- Next Action: **CallCentral (A9)**
  - Time Slot Assigned: 80 – 90
- Next Action: **Move (A9, B9)**
  - Is it a Repair Action? : Yes
  - Meet all deadlines? : **No** (Agent \( \alpha \) needs A9 at 85)
  - Backtrack to action **Move(A8, A9)**
    - Find another Time Slot
    - New Time Slot: **110 – 115** (Valid Time Slot)
Plan Merging Algorithm

Case 4: All the effects of CA are already achieved

**WHAT TO DO?**

- Mark CA as redundant

**POST PROCESSING**

- Remove all redundant actions from the plan
- Recursively remove all actions which achieve only the preconditions of removed action
EXAMPLE

- **Current Action:** OpenDoor (C11)
  - Returned Time Slot: 172 - 175
  - Redundant(OpenDoor(C11)) \(\iff\) true
    - *Because* openedDoor(C11) is true at time 172

- When the plan is returned
  - Remove OpenDoor(C11) from plan
  - Also remove TakeKey(C11, key1) from plan
Plan Merging Algorithm

Case 5: Action CA’s preconditions can not be achieved

- Remove action CA from the plan and compute a plan to achieve effects of CA

  - I = State just before CA
  - G = Effects (CA)

- Plan should have no conflict with Reactive Plan $P_\alpha$ and if CA is a repair action, repair effects must meet their deadline

  - ReplacementPlan = Coordinated-Sapa (I, G, $P_\alpha$)

- If a plan is returned, replace the removed actions with the plan

- If a deadline is violated, backtrack to the threat action

- If no plan possible, then remove another action CA + 1

  - G = G U Effects (CA + 1) \ Pre (CA + 1)
Plan Merging Algorithm

**EXAMPLE**

- Current Action: TakeEnergy(B13, energy1)
- Preconditions can not be achieved
  - Repair the plan
Plan Repair Algorithm

- Create a CPP by removing `TakeEnergy(B_{13}, \text{energy}_1)`
  - \( I = \{ \text{at}(\beta, B_{13}), \text{at}(\text{energy}_1, B_{13}), \text{at}(\text{energy}_2, B_{13}) \} \)
  - \( G = \{ \text{hasEnergy}(\beta, \text{energy}), \text{at}(\beta, B_{13}) \} \)

- Call Coordinated-\textit{Sapa} to solve this CPP
  - Coordinated-\textit{Sapa} returns fail
  - \textit{Why?} \text{energy}_2 \text{ is also needed by Agent } \alpha
Plan Repair Algorithm

- Create another CPP by removing \text{Move}(B_{13}, A_{12})
  
  \[ I = \{ \text{at}(\beta, B_{13}), \text{at}(\text{energy}_1, B_{13}), \text{at}(\text{energy}_2, C_{15}) \} \]
  \[ G = \{ \text{at}(\beta, A_{12}) \} \]

- Call \text{Coordinated-Sapa} to solve this CPP
  - A plan is returned to enter A_{12} by taking the fuel from D_{14}

**POST PROCESSING**

- This plan will become a replacement for both \text{TakeEnergy}(B_{13}, \text{energy}_1) and \text{Move}(B_{13}, A_{12})
P-CLAIM: AOP Language supporting Temporal Planning
P-CLAIM

- Extension of CLAIM [Suna and El Fallah Seghrouchni 2005]
- An AOP language having:
  - Cognitive aspects specific to intelligent agents
  - Communication primitives
  - Mobility primitives
  - Temporal planning capability (New)

- P-CLAIM Agent:
  - Is autonomous, intelligent and mobile
  - Has a mental state containing knowledge, goals, and capabilities
  - Is able to communicate with other agents
  - Entails a planning based behaviour (New)
  - Achieves goals based on their priorities (New)
  - Maintains the stability of the plan in the dynamic environments (New)
Agent Definition in P-CLAIM

- Similar to CLAIM, but
  - Added priorities to goals
    - High Preemptive (reactive): Should be immediately planned for and achieved
    - High and Normal (proactive): High priority goals should be achieved before normal priority goals
  
- Capabilities in CLAIM $\rightarrow$ (Activities + Actions) in P-CLAIM
  - Activities are short plans to achieve tasks
  - Actions are primitive tasks with duration
Example: Analyser Agent Code

defineAgent Analyzer{ 
    authority=null;
    parent=null;
    knowledge=
    {
        agent_in(a1);hasMobile();
        connected(a1,a2);connected(a2,a3);connected(a3,a4);
        connected(a4,a5);connected(b1,b2);connected(b2,b3);
        connected(b3,b4);connected(b4,b5);
        keyPlaced(door1,c3);doorBetween(b4,b5,door1);closed(door1);
    }
    goals={analyze_cell(b5,PDA1);}
    messages=null;
    activities=
    {
        MoveToCell1{
            if the agent is already in the goal cell, do nothing
            message=moveToCell(?c);
            condition=hasKnowledge(agent_in(?c));
            do{}
            effects=hasKnowledge(agent_in(?c));
        }
    }
MoveToCell2{
  if the agent is in the neighbouring cell, move to the
  goal cell in a single step
  message=moveToCell(?c);
  condition=and(hasKnowledge(agent_in(?b)),
                hasKnowledge(connected(?b,?c)));
  do{send(this,OP_move(?b,?c))}
  effects=hasKnowledge(agent_in(?c));
}
Agent Life Cycle
1- Fetch goals one by one from GRG and GPG and calls Compute_Plan to compute a plan for the goal.

2- GPG Goals are accessed only when GRG is empty.

3- Sends a suspension signal to Executor if the goal is reactive i.e. from GRG.
Plan Computation

- JSHOP2 algorithm [Nau et al. 2003] is used to compute a totally ordered plan for each goal

- An HTN planning algorithm

- Decomposes the task into sub-tasks by applying methods

- Recursively applies the same procedure on every composite sub-task until there are only primitive tasks
Temporal Converter

• Input to procedure
  • A totally ordered plan
  • Actions’ information (Add, Del, Pre, Durations)

• Output of procedure
  • A position constrained parallel plan
    • Every action is assigned a time stamp
    • Multiple actions can possibly lie in parallel
Merging the New Plan to Global Plan
Merging the New Plan to Global Plan

Planner

Reactive Goal

Merge at the start of $P_{\text{exec}}$

Proactive Goal

Append at the end of $P_{\text{exec}}$

Plan Under Execution ($P_{\text{exec}}$)
Summary

- Coordinated Planning Problem
  - Computing plan while coordinating it with another plan
  - Sapa $\rightarrow$ Coordinated-$Sapa$

- Proactive-Reactive Coordination Problem
  - Modifying a plan to remove conflicts with a higher priority plan $\rightarrow$ Plan Merging Algorithm

- An agent oriented programming language supporting:
  - Temporal Planning
  - Plan Repairing
  - Dealing with different priority goals
Summary

- Properties of the temporal conversion algorithm
  - Soundness (Proof: By construction)
  - Termination
- Properties of the plan merging algorithm
  - Soundness (Verified by experimental evaluation)
- Computational Complexity
  - Temporal Converter: Quadratic
  - Plan Merging Algorithm:
    - Worst Case: Exponential
    - Average Case: Quadratic
Perspectives

- Coordination of plans for same priority goals
  - Negotiation based strategy

- Planning with incomplete information
  - Information gathering actions

- Improvement in the computational complexity of plan merging algorithm
  - Propose efficient heuristics to reduce the number of backtracks
Presentation outline

- Introduction
  - Multi-Agent Context
  - (Multi-Agent) Temporal planning
  - Multi-Agent Planning & Coordination
- Framework 1: Temporal Planning and the Plan Coordination Mechanisms
  - Coordination during planning
  - Coordination after planning
  - P-CLAIM
- Framework 2: Overview of temporal planning based on Hybrid Automata
  - Tactical aircraft simulation
  - Our approach
- Conclusion and perspectives
Modeling, Control and Validation of Multi-Agent Plans
Application to the Aircraft Simulation Domain

Collaboration with
Irène Degirmenciyian-Cartault (Dassault Aviation)
Frédéric Marc (LIP6 / Dassault Aviation)
An Interception Mission

- Take Off order
- Interception Data
- CAP
- Guided Flight
- Localization
- Triangulation
- Shoot
- Back to Base
- Reorganization

Need of reorganization and/or replanning when new events occur
Framework requirements

- Collective behavior is expected in open environment
  - Temporal planning, re-planning and synchronization (collective actions)
- Management of continuous and discrete Resources evolving with different speeds
  - e.g. kerosene, time, number of missiles, etc.

⇒ formalism enabling the representation of all this features
⇒ mechanisms for multi-Agent Planning and on-line Re-planning
Definition of an hybrid automaton [HEN 96]

- \( A = \langle Q, E, Tr, H, q_0, l \rangle \) :
  - \( Q \) is the set of reachable states,
  - \( E \) is the set of labels,
  - \( Tr \) is the set of edges,
  - \( H \) a set of clocks,
  - \( q_0 \) is the initial location,
  - \( l \) the application associating the elements of \( Q \) with elementary properties verified in this state.
Cycle for Building Feasible Multi-Agent Plans

- **Allocation Module**: Validation: Coherent allocation of tasks (functionnal constraints)
- **Planning Module**: Validation: level of ressources
- **Synchronisation Module**: Validation: feasibility & temporal constraints

*Modeling of the individual plans by means of hybrid automata*
*Modeling of the multi-agent plans by means of a network of synchronized hybrid automata*

**Graph 2**

Replanning by means of operators of planning
Partially Ordered Set of Tasks

SFP: Local Constraints Propagation

Individual Planning

Set of Individual Hybrid Automata:
Individual plans locally feasible (Feasible Paths)

Set of Synchronized Hybrid Automata

Network of Dependencies between the agents

Set of Feasible Multi-Agent Plans

Graph

Event

ALLOCATION

SYNCHRONISATION
Individual Planning

- Modeling of individual plans by means of Hybrid automata
  - An extension of the timed automata [ALU 94] in which different clocks can evolve with different speeds.
  - Take into account different resources (discrete and continuous) and temporal constraints

\[
\begin{align*}
T & \quad \text{label} \\
1 & \quad h_1 = 1 \land h_2 < 4 \\
2 & \quad h_1 := 0 \land h_2 = h_2 + 4
\end{align*}
\]
Individual planning

Individual Planning algorithm:
recursive algorithm providing from a set of partially ordered tasks all the alternatives of actions

3 possible plans

Plains modeled as Hybrid Automata
Synchronisation of HA

- Cartesian Product of a set of hybrid automata
1. The Reachable States
   To model a duration, each task of Ta must be divided into a **start state** and an **end state**.
   For each $t_i \in Ta$, and for $i \in [1, \text{card}(Ta)]$,
   \[
   Q = \bigcup \{ t_i, \text{start, } t_i, \text{end} \}
   \]

2. The Labels of Transition
   Two types of transition : $C_{in}$ and $C_{ext}$
   - $C_{in}$: internal conditions enable moving from a start state to an end state: task execution or interruption
   - $C_{ext}$: external conditions enable moving from a task to another

   \[
   E = C_{in} \cup C_{ex}
   \]
Modeling of an Individual Plan with Hybrid Automata

3. The Clocks

Represent the variables of the agents: the time and the resources

Assumption: The local clocks are synchronized on a global clock

Operations on clocks: use, consume, and produce

For each $t_i \in T_a$, and for $i \in [1, \text{card}(T_a)]$, $\{\text{Res}_{t_i}\}$ represents the set of resources used, consumed or produced, with the task $t_i$

$$H = \bigcup \{\text{Res}_{t_i}\}$$

4. The initial state

Initial level of the different resources
Modeling of an Individual Plan with Hybrid Automata

5. The Edges
Two types of edges: \( \text{in}_{\text{edge}} \) and \( \text{ex}_{\text{edge}} \)

- \( \text{in}_{\text{edge}} \)
  Considering \( tr \in \text{in}_{\text{edge}} \), \( s \in Q_{\text{start}} \) and \( s' \in Q_{\text{end}} \)
  \[ tr = <s, (X_{\text{interrupt}}, l_x, X_{\text{post}}), s'> \]

- \( \text{ex}_{\text{edge}} \)
  Considering \( tr \in \text{ex}_{\text{edge}} \), \( s \in Q_{\text{end}} \) and \( s' \in Q_{\text{start}} \)
  \[ tr = <s, (X_{\text{pre}}, l_x, -), s'> \]
Guided Flight \([?t0, t1]\)  

Required resources:
- Kerosne: \(k\k\)
- Time: \(t\)

Duration: \(dN\)

Precondition: \(k > X + \text{consume}(k + dN)\);

Interruption: \(k > X \land t < t1\);

Post: \(k := k - \text{consume}(k, dN)\)

Propagation of constraints:

- \(k > X + \text{consume}(k + dN) \land t < t1 - dN\)
- \(ker > X \land t < t1\)
- \(d\)
- \(ker := ker - \text{consume}(ker, dN)\)
Individual planning: Module of control
[AAMAS 2004, IAT 2004]

- Control of the individual plans execution
  - Control of the agents resources (execution constraints of plans)
  - Taking into account various possible evolutions

Unfolding of automata: set of feasible paths (FP: Feasible Path)

![Automaton diagram](image)

- FP 1
- FP 2
- FP 3

Set FP:

- Set FP 1
- Set FP 2
- Set FP 3
Control of Plans

Automaton

UNFOLDING

Set of possible paths

SFP

FP 1

FP 2

FP 3

feasible path: FP

Preponderant constraints on a state

Different sets of constraints for one task

To Validate a set of feasible paths

=  

To validate on set of preponderant constraints
Planification individuelle : Réduction des plans

Si violation de contraintes fonctionnelles, ressources :
Etat d'instabilité => Etat de stabilité par réduction des SFPs

=> Mise à jour des SFPs et des plans multi-agents

Respect des contraintes
R-Compatibilité
Reduction of SFP

Stable SFP
- Constraints on resources OK
- Multi-agent plan OK

Reduction of the SFP:

Pruning of the no more feasible paths

Unstable SFP
(constraints violation, etc..)

=> Converge to only one feasible path
Extension to MAP (1/3)

- Each agent has the knowledge of the initial global plan and has to synchronize its own plan with the others according to the graph of dependencies.

**Constraints of coordination and execution**

- Multi-agent plan is a network of synchronized hybrid automata: synchronization techniques (send/reception messages, shared variables, etc...)

**Representation by means of a synchronized product of hybrid automata**
Extension to MAP (2/3)

- An example: the $S_{\text{start}}$ Connector

Considering $S_{XYZ} / S_{XYZ} = \{Z \text{ Seq } X, Z \text{ Seq } Y, X \text{ } S_{\text{start}} Y\}$ and two agents A1 and A2 such as A1 is in charge with the tasks (Z, X) and A2 is in charge with (Y).

- $S_{XYZ}$ for A1/ $S_{XYZ, A1} = \{Z \text{ Seq } X, Z \text{ Seq } Y^*, X \text{ } S_{\text{start}} Y^*\}$
- $S_{XYZ}$ for A2/ $S_{XYZ, A2} = \{Z^* \text{ Seq } Y, X^* \text{ } S_{\text{start}} Y\}$

Representation in a graph of functional dependencies
Extension to MAP (3/3)

Incidence of the operators on the definition of the individual automata:

- Extension of the set of the reachable states
- Extension of the set of the edges

Without the constraints of the graph:

Individual automata

- Automaton for $A_1$
  - init $\rightarrow Z_1 \rightarrow Z_2 \rightarrow X_1 \rightarrow X_2$
- Automaton for $A_2$
  - init $\rightarrow Y_1 \rightarrow Y_2$

With the constraints of the graph:

Individual automata once synchronized

- Automaton for $A_1$
  - init $\rightarrow Z_1 \rightarrow Z_2 \rightarrow Es \rightarrow Sync_1 \rightarrow X_1 \rightarrow X_2$
  - respSynchro $\rightarrow$ mess
- Automaton for $A_2$
  - init $\rightarrow Es \rightarrow Sync_1 \rightarrow Y_1 \rightarrow Y_2$
  - Sync_2 $\rightarrow$ mess
  - Sync_1 $\rightarrow$ mess
  - respSynchro $\rightarrow$ mess
A multi-agent example: an Interception Mission Scenario

- 4 agents composing a Division
- Graph triggered in response to the arrival of a new threat in the friend area

Interception graph
A multi-agent example: an Interception Mission Scenario

Agent 1

Agent 2

Agent 3

Agent 4
The SCALA Layout

Events

Tasks & constraints

Roles & cooperation schemas

Abstraction level

SCALA

JACK™

JAVA

BDI Agents

Object programming
Summary

- Formal model for multi-agent temporal planning
  - Feasible multi-agent plans taking into account functional constraints, resources and goals
  - On-line re-planning
  - Validation of feasible MAP based on the Model Checker HYTECH (reachability and synchronizations)

- Applicability
  - Efficient modeling of complex systems
  - Abstraction of physical models: separation between reasoning and physical models
  - Easy integration of new scenario without recompilation
Conclusion and perspective

- Reasoning about actions with durations
  - Actions with durations in the first framework
  - Time is one resource among others in the second

- Perspective...
  - Complexity and scalability problem
    - Use timeslot to reduce the planning complexity?
    - Take benefit from MAS to distribute de coordination
  - These models respond well in simulation, but what about real–life applications?
Publications

Publications

- "SCALA : une approche Multi-Agent pour la conception de Systèmes Complexes - Application à la simulation de missions aériennes" I. Degirmenciyan-Cartault et F. Marc, JFSMA'02
- "Multi-agent planning as a coordination model for complex Self-organized systems" F. Marc, I. Degirmenciyan-Cartault et A. El Fallah-Seghrouchni IAT'03, IEEE
- "An integral cycle for building feasible multi-agent plans" F. Marc, I. Degirmenciyan-Cartault et A. El Fallah-Seghrouchni IAT'04, IEEE
- "Modeling, Control and Validation of Multi-Agent Plans in Dynamic Context" A. El Fallah-Seghrouchni, I. Degirmenciyan-Cartault et F. Marc AAMAS'04, ACM
"On dit que le temps change les choses, mais en fait le temps ne fait que passer et nous devons changer les choses nous-mêmes"

Andy Warhol

Merci pour votre attention