Planning in Multiagent Systems

Mathijs de Weerdt

Cees Witteveen

May 18, 2007
Goals of this tutorial

• Broad view on multiagent planning problems and multiagent planning techniques

• A bit background knowledge on AI planning techniques and their relevance for multiagent planning

• Some examples of techniques to coordinate plans
Tutorial Contents

1. Introduction & taxonomy

2. Single-agent planning techniques for multi-agent planning

3. Details of some approaches to coordinate plans
   1. Pre-planning coordination
   2. Coordination during planning (GPGP)
   3. Post-planning coordination (plan merging)
Overview of first part of tutorial

- Introduction to multiagent planning
  - Relation with multiagent systems
  - Relation with planning
- Taxonomy of multiagent planning problems
- Taxonomy of multiagent planning techniques
- Discussion
Why planning in multiagent systems?

- More efficient system performance on run-time
  - Come prepared
  - Prevent deadlock
  - Lower costs
  - Accomplish task more quickly
- Useful assignment of resources, use of capabilities
Multiagent Planning (MAP) and Multiagent Systems (MAS)

- MAS: Coordination of autonomous entities
- MAP $\subseteq$ MAS (hence this tutorial)
- MAP: Focus on
  - Coordination of actions before execution
  - Finding correct actions to attain goals
- Strongly related to task allocation and auction/negotiation techniques
MAP and AI planning

- MAP $\supseteq$ AI Planning for multiple agents
  - Execution in parallel (instead of sequentially)
    $\rightarrow$ Parallel plans

- MAP $\supseteq$ Planning by multiple agents (distributed)
  - Incoherent plans $\rightarrow$ need for coordination; more difficult
  - Why then?
    - Privacy & autonomy
    - No central point of failure
    - Local $\rightarrow$ more efficient reaction on incidents (when communication limited) & no central point of failure
    - Speed-up: in parallel & smaller problems
Applications of Multiagent Planning

- Planetary explorations
- Multi-player video games
- Taxi companies
Planetary explorations

- Dependent?
- Communication?
- Cooperative?
- Resolving conflicts or improving efficiency?
- Incident rate?
Planetary explorations

- More or less independent
- Communication difficult and costly
- Cooperative
- Improving efficiency
- Incidents well possible
Multi-player video games

- Self-interested players
- Independent
- Potential for coordinated behavior
Taxi companies
Outline from here on

• Introduction to multiagent planning
  • Relation with multiagent systems
  • Relation with planning

• Taxonomy of multiagent planning problems

• Taxonomy of multiagent planning techniques

• Discussion
Taxonomy of coordination

Coordination

Cooperation

Planning

Centralized Planning

Competition

Negotiation

Distributed Planning

Multiagent Planning

From: M. Huhns, L. Stephens, Ch.2 from G. Weiss (1999)

Multiagent systems: A modern approach to Distributed Artificial Intelligence
Taxonomy of MAP problems

Many ways to look at multiagent planning problems

• Strongly related → Independent
• Cooperative → Self-interested
• Resolving conflicts → Exploit efficiency
• No communication → Reliable communication
• Incident rate
Strongly related → Loosely coupled → Independent

Strongly related because
- Joint actions
- Limited shared resources

Requires crisp coordination

Examples
- Lift a box together
- Car assembly
- Robocup
- Hospital
- PhD research
Cooperative → Self-interested

Independent

Cooperative

Rovers

Robocup

Lifting boxes

PhD research

Hospital

Human soccer

UN-operation

Strategy game

Taxi companies

Traffic

Supply chains

Self-Interested / Private

Strongly related
<table>
<thead>
<tr>
<th>Traffic</th>
<th>Strategy game</th>
<th>PhD research Rovers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hospital</td>
<td>Robocup</td>
<td>Taxi companies</td>
</tr>
<tr>
<td>Lifting boxes</td>
<td>Supply chains</td>
<td></td>
</tr>
<tr>
<td>Resolve conflicts</td>
<td>Both</td>
<td></td>
</tr>
<tr>
<td>Strongly related</td>
<td>Exploit efficiency</td>
<td></td>
</tr>
<tr>
<td>Independent</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Resolving conflicts → Exploit efficiency
No communication → reliable communication

Examples:
• Rescue Robots
• Planetary explorations
• Military operations
Incident rate

- Any system that is used should deal with unexpected events
  - Plan adaption (repair) or
  - Start from scratch (replanning)
- Multiagent planning can help here by locally replanning / plan repair
  - when no communication possible
  - to reduce the impact of incidents
Overview of this part of tutorial

• Introduction to multiagent planning
• Taxonomy of multiagent planning problems
• Taxonomy of multiagent planning techniques
• Discussion
Taxonomy of MAP techniques

- Where are plans created?
- When coordination in MAP process?
- How are plans coordinated?
Where are plans created?

- Centralized
- Distributed
- Partially distributed
- Distributed for a centralized plan
Where are plans created?

Centralized
- Optimal (potentially)
- Communication only twice (before & after)

Algorithm:
1. Label actions with agent names
2. Plan
3. Decompose into sub-plans
4. Add synchronization actions
Where are plans created?

Distributed

• Reduce computation time
• Keep privacy (potentially)
• Scalable
• Execution and control are also distributed

Examples:

• Ephrati (1995): by plan merging
• vd Krogt (2005): by plan repair
**Where are plans created?**

**Partially distributed**
- Share parts of your plan

**Examples:**
- Corkill (1979): Distributed NOAH (shared world model)
- Durfee, Decker, Lesser (1986-) Partial global planning (shared plan)
Where are plans created?

Distributed for a centralized plan
  • Specialized planning agents

Examples:
  • Kambhampati (1991): Combining specialized reasoners and planning
  • Wilkins (1998): Multiagent planning architecture
When coordination in MAP process?

1. Global task refinement  
   = AI planning
2. Task allocation
3. Coordination before planning
4. Individual planning
5. Coordination after planning
6. Plan execution
When coordination in MAP process? (pre)

3. Coordination before planning
   (pre-planning coordination)
   - Social laws (e.g. traffic rules): Shoham & Tennenholtz (1992)
   - Derive specific constraints for agents: Buzing, Valk et al. (2006)
When coordination in MAP process? (during)

4. Individual planning
   Coordination during planning
   • Distributed NOAH (Corkill, 1979)
   • Partial global planning (Decker, 1992)
   • Through plan repair (vd Krogt, 2005)
When coordination in MAP process? (post)

5. Coordination after planning
   • Plan merging (Ephrati, 1993)

6. Coordination during (plan) execution
   • When communication is unreliable (Koes, 2006)
   • “Normal” MAS/distributed systems solutions
     • FIFO-queues
     • Semaphores
When coordination in MAP process? (continual)

- Plans are being executed during planning and coordination
- May break and re-make commitments
  - unexpected event/failure
  - goal change

(DesJardins, 2000)

Distributed continual planning

- Task refinement
- Task allocation
- Pre-planning coordination
- During-planning
- Post-planning coordination
- Execution
Difficulties of continual planning

- Chain reactions of changes
- Cyclic dependencies
How are plans coordinated?

What are the main coordination problems?

- Maintenance of dependencies?
  - Distributed check for cycles
  - Shared global plan
  - Before / Afterwards

- Distribution of tasks and resources?
  - Contract net protocol
  - Auctions (combinatorial?)
  - Negotiations, etc.
Taxonomy of MAP

Problems:
- Strongly related → Independent
- Cooperative → Self-interested
- Resolving conflicts → Exploit efficiency
- No communication → Reliable communication
- Incident rate

Techniques:
- **Where** are plans created?
  - Centralized → Distributed
- **When** coordination in MAP process?
  - Before planning → During planning → Post-planning
- **How** are plans coordinated?
  - Dependency checks?
  - Task/resource allocation?
Where are we?

- Introduction to multiagent planning
- Taxonomy of multiagent planning problems
- Taxonomy of multiagent planning techniques
- Discussion
Recent focus

- Mixed initiative multiagent planning
  - Using domain knowledge (in task networks)

- More application-oriented work:
  - search & rescue
  - military operations ("Coordinators")
  - logistics
Discussion: Reasons for not doing multiagent planning

- Privacy & autonomy
  - First negotiating, then (optimal) planning

- No central point of failure
  - Just introduce redundancy

- Limited communication (on execution)
  - Contingent planning on forehand & with constraints on-line

- Speed-up: in parallel & smaller problems
  - Use a grid and parallel computing techniques
Recommended reading

- Durfee’s chapter (3) on Distributed Problem Solving and Planning in Weiss (1999), *Multiagent systems: A modern approach to Distributed Artificial Intelligence*


- (Links to) material can be found on: [http://www.st.ewi.tudelft.nl/~mathijs/tutorial.php](http://www.st.ewi.tudelft.nl/~mathijs/tutorial.php)
Other references

Tutorial Contents

1. Introduction & taxonomy

2. Single-agent planning techniques for multi-agent planning

3. Details of some approaches to coordinate plans
   1. Pre-planning coordination
   2. Coordination during planning (GPGP)
   3. Post-planning coordination (plan merging)
From AI planning to multi-agent planning
Aim of AI

• “Intelligent” systems, decide themselves
  • what to do, and
  • how to do it.

• Planning is the general problem of...
  • given what to do (goals),
  • determine how (and when) to do it (plan).

• Currently planning research community very active
  • Significant scale-up
  • Bi-annual planning competition
Contents

- Introduction to AI planning
  - Applications
  - Problem
- Classical planning
  - Model
  - STRIPS
- Refinement planning framework
  - Partial plans
  - Plan space refinement
  - HTN planning
- Complexity of planning
Applications

- Action choice + resource handling
  - for transportation of goods
  - at schools, hospitals
  - Hubble Space Telescope scheduler

- Interactive decision making
  - for military operations
  - for astronomic observations
  - Plan-based interfaces (plan recognition)
Planning problem

• **How** to get from the current state to your goal state?

• Planning involves
  • Action selection
  • Action sequencing
  • Resource handling

• Plans can be
  • Action sequences
  • Policies/strategies (action trees)
Additional difficulties

Because the world is uncertain:
• Dynamic
• Stochastic
• Partially observable

And because actions
• take time
• have continuous effects
AI Planning background

- Focus on **classical planning**; assume none of the above
- Deterministic, static, fully observable
  - “Basic”
  - Most of the progress up to 2005/2006
  - Ideas often also useful for more complex problems
Contents

• Introduction
  • Applications
  • Problem

• Classical planning
  • Model
  • STRIPS

• Refinement planning framework
  • Partial plans
  • Plan space refinement
  • HTN planning

• Complexity of planning
Classical planning model

Origins:
- STanford Research Institute Problem Solver (’71)
- derived from GPS = human problem solving (’61)

States described by propositions currently true

Actions: general state transformations described by sets of pre- and post-conditions

Represents a state-transition system (but more compact)
Planning is searching

...in **state space** (or...)

Choose between possible actions

- Depth-first
- Breadth-first

I want to be here!

**initial state**

a plan is a route in state space
Searching for a plan

start from initial state, try all possible actions
→ large search space

STRIPS: regression: look at goal state!

I’m here!
STRIPS Formalism (now in PDDL)

- action: preconditions, add, delete effects
- pickup(B1, B2)
  - precondition: empty & clear(B1) & on(B1, B2)
  - add-effect: holding(B1), clear(B2)
  - delete-effect: empty, on(B1, B2), clear(B1)
- problem: initial state, actions/operators, goal description
- objects and variables
**STRIPS algorithm**

STRIPS( s, g )

returns: a sequence of actions that transforms s into g

1. Calculate the difference set \( d = g - s \).
   1. If \( d \) is empty, return an empty plan
2. Choose action \( a \) whose add-list has most formulas contained in \( g \)
3. \( p' = \text{STRIPS}( s, \text{precondition of } a ) \)
4. Compute the new state \( s' \) by applying \( p' \) and \( a \) to \( s \).
5. \( p = \text{STRIPS}( s', g ) \)
6. return \( p';a;p \)
## STRIPS demo

(by CI – space, British Columbia (CA))

<table>
<thead>
<tr>
<th>Action</th>
<th>Preconditions</th>
<th>Add List</th>
<th>Delete List</th>
</tr>
</thead>
<tbody>
<tr>
<td>pickup(B1, B2)</td>
<td>empty &amp; clear(B1) &amp; on(B1, B2)</td>
<td>holding(B1), clear(B2)</td>
<td>empty, on(B1, B2), clear(B1)</td>
</tr>
<tr>
<td>pickutable(B)</td>
<td>empty &amp; clear(B) &amp; ontable(B)</td>
<td>holding(B)</td>
<td>empty, ontable(B), clear(B)</td>
</tr>
<tr>
<td>putdown(B1, B2)</td>
<td>holding(B1) &amp; clear(B2)</td>
<td>empty, on(B1, B2), clear(B1)</td>
<td>clear(B2), holding(B1)</td>
</tr>
<tr>
<td>putdowntable(B)</td>
<td>holding(B)</td>
<td>empty, ontable(B), clear(B)</td>
<td>holding(B)</td>
</tr>
</tbody>
</table>
STRIPS planner

- Questions?
  - Optimal?
  - Sound?
  - Complete?
Classical planning in a multiagent setting

- When an agent is unable to do a task
  - Give task to other agent
  - Other: adapt current plan → plan “repair”

Van der Krogt (2005)
Contents

• Introduction
  • Applications
  • Problem

• Classical planning
  • Model
  • STRIPS

• Refinement planning framework
  • Partial plans
  • Plan space refinement
  • HTN planning

• Complexity of planning
Refinement planning framework

- Framework to capture all planning algorithms
- Idea: Narrow set $P$ of potential action sequences
- Specify action sequences by partial plans
- Subcontents:
  - Generic template
  - Refinement strategies
Refinement planning template

Refineplan( $P$ : Plan set)

1. If $P$ is empty, Fail.
2. If a minimal candidate of $P$ is a solution, return it. End
3. Select a refinement strategy $R$
4. Apply $R$ to $P$ to get a new plan set $P'$
5. Call Refineplan($P'$)

Termination ensured if $R$ complete and monotonic.
Existing Refinement Strategies

- State space refinement: e.g. STRIPS
- Plan space refinement: e.g. Least commitment planning
- Task refinement: e.g. HTN
Plan space refinement (I)

- Least commitment planning (Weld, 94)
  - search in plan space instead of state space
  - represent plans more flexible: not a sequence, but a partially ordered set
  - keep track of decisions and the reasons for these decisions
Partial Plans: Syntax

Partial plan = (Actions, partial Ordering, causal Links)

- causal links = Interval preservation constraint (IPC) \((a_3, q, a_4)\)
  - q must be preserved between \(a_3\) and \(a_4\)
Plan space

A state in the (plan) search space is a partial plan (instead of a description of the state of the world)
• Start with
  • null (empty) plan
  • agenda
    = list of (precondition, action) goals
    = \{(g_1, a_\infty), (g_2, a_\infty), (g_3, a_\infty), \ldots\}

• Deal with one (g,a) at a time
POP example

\[ \{ \text{on}(b_1, b_2), \text{on}(b_2, b_3) \} \]

\begin{align*}
a_1: & \text{ move } b_2 \text{ from table to } b_3 \\
\neg \text{clear}(b_2): & \text{ move } b_1 \text{ from table to } b_2 \\
\text{clear}(b_2): & \\
\{ \text{on}(b_3, b_1), \text{clear}(b_2), \\
\text{on}(b_1, \text{table}), \text{on}(b_2, \text{table}) \} \\
a_0: &
\end{align*}
POP( (A,O,L), agenda )

1. **Termination**: if agenda is empty return (A,O,L)

2. **Goal selection**: select a \((g,a_{\text{need}})\) from the agenda

3. **Action selection**: choose an action \(a_{\text{add}}\) that adds \(g\). Update \(L, O,\) and \(A\)

4. **Update goal set**: remove \((g,a_{\text{need}})\) from agenda, and add its preconditions \((..., a_{\text{add}})\)

5. **Causal link protection**: for every action \(a_t\) that might threaten a link, add an ordering constraint
Tradeoffs among Refinements

State space refinement:
• commit to both order and relevance of actions
• include state information (easier plan validation)
• lead to premature commitment
• too many states when actions have durations

Plan-space refinement:
• commit to actions, avoid constraining order
• increase plan-validation costs
• reduce commitment (large candidate set /branch)
• easily extendible to actions with duration
Partial plans in a multiagent setting

- Broadcast (abstraction of) part of your plans relevant for others → “partial global plan”
- Keep updating this global plan until nothing changes

Generalized Partial Global Planning (Durfee, Decker, Lesser, et al.)
Where are we?

- Introduction
  - Applications
  - Problem
- Classical planning
  - Model
  - STRIPS
- Refinement planning framework
  - Partial plans
  - Plan space refinement
  - HTN planning
- Complexity of planning
HTN Planning

Problem reduction

• Decompose tasks into subtasks
• Handle constraints (e.g., taxi not good for long distances)
• Resolve interactions (e.g., take taxi early enough to catch plane)
• If necessary, backtrack and try other decompositions

travel(Delft, Honolulu)

travel(Delft, AMS)
airport(Delft,AMS)
get taxi
fly(AMS,HNL)

travel(HNL, Honolulu)
airport(Honolulu,HNL)
get taxi

Hierarchical multiagent planning

- Task structure for all involved agents
  - QAFs
    - SUM (\sim \text{AND}), SyncSUM
    - MAX (\sim \text{OR}), MIN
  - Non-local effects
    - Enables, Disables
    - Facilitates, Hinders
- TAEMS (Decker), c_TAEMS for Coordinators (Boddy et al.)
- Zlot and Stentz (2006)
Complexity of planning

- PLANEX: plan existence problem is PSPACE-complete for propositional STRIPS
- Restrictions on preconditions and effects allowed helps to reduce complexity
Complexity of planning

PLANEX is PSPACE-complete
- plan length is at most \( l = 2^n \)
- hardness: each PSPACE problem can be translated by making Turing-machine operations into actions
- plan existence with non-deterministic Turing machine in \( O(\log l) \) space (polynomial in \( n \))

\[
\text{plan}(s,g)
\]
1. if \( s = g \) return true
2. guess intermediate state \( s' \)
3. return \( \text{plan}(s,s') \) and \( \text{plan}(s',g) \)
Complexity of planning

pre: preconditions

effs: effects

* pre * effs

1 pre * effs

2 +pre 2 effs

* pre 1 effs

* pre * +effs

1 pre 1 effs

* pre * effs

1 pre 1 effs

1 +pre 1 effs

1 pre * effs k goals

0 pre * effs

PSPACE-complete

NP-complete

Polynomial
Apply other planning techniques to multiagent planning:

- Non-deterministic actions (surprises)
- Partially observable world (exact costs unknown)
- Durative actions (move, travel) and continuous variables (capacity, fuel, distance, time) $\rightarrow$ optimality
- Preferences instead of goals
Recommended reading


Other references

Tutorial Contents

1. Introduction & taxonomy

2. Single-agent planning techniques for multi-agent planning

3. Details of some approaches to coordinate plans
   1. Pre-planning coordination
   2. Coordination during planning (GPGP)
   3. Post-planning coordination (plan merging)
Details of some approaches to coordinate plans
Overview of third part of this tutorial

Focus on this part
- planning
- coordination

Joint plan execution
Pre-planning coordination

coordination ends before planning starts; guarantees coordinated individual plans

examples: social laws, social conventions, pre-planning protocols

ensures autonomous planning

independent planning systems

combining individual plans should be trivial and always successful.
Coordination during planning

coordination and planning as intertwined processes; coordination is applied to partial plans

agents do not plan autonomously but interact during planning

example: (G)PGP, COABS
Post-planning coordination

- joint plan execution
- tasks
- agents
  - task refinement
  - task allocation
  - planning
    - Agent 1
    - Agent 2
    - Agent 3
    - Agent 4
  - independent planning systems
  - post planning coordination
  - improves quality of agent plans

exploits positive interactions, removes negative interactions (conflicts)
examples: plan integration, plan merging, plan fusion
**General framework**

- **Agents**
  - with different capabilities

- **Partially ordered set of tasks**
  - where tasks require different capabilities

- **Task allocation**

- **Dependencies between agents**
General framework

- Each agent wishes to plan autonomously.
- Plan $P_i$ induces additional ordering constraints on the tasks assigned to agent $A_i$.

In general, autonomous planning can cause *deadlocks* upon execution (cyclic dependencies).

Note that:
- every plan $P_i$ is feasible and satisfies the local constraints;
- there is no feasible joint plan respecting the individual plans.
Pre-planning Coordination Problem

Given
- a set $T$ of tasks,
- a set of constraints $C$, and
- a partitioning $[T_1, T_2, \ldots, T_k]$ of $T$ over agents $1, \ldots, k$ such that $C_i \subseteq C$ are the constraints induced by $T_i$.

How to ensure that every feasible plan $P_i$ constructed by $A_i$ and satisfying $(T_i, C_i)$ can be used to constitute a joint plan $P$ for $(T, C)$ without revising it?
Multi-modal logistic problems

- agent 1
  - city
  - airport

- agent 2
  - flight agent

- agent 3
  - order
  - sequence of elementary tasks
Logistic problem: set-up

Complex task = set of orders

- loc12 - loc10 - loc20 - loc24
- loc32 - loc30 - loc20 - loc21
- loc12 - loc10 - loc30 - loc33
- loc21 - loc20 - loc30 - loc31
Logistic problem: set-up

orders as dependent tasks

<table>
<thead>
<tr>
<th>t</th>
<th>&lt; t</th>
<th>&lt; t</th>
</tr>
</thead>
<tbody>
<tr>
<td>12-10</td>
<td>10-20</td>
<td>20-24</td>
</tr>
<tr>
<td>32-30</td>
<td>30-20</td>
<td>20-21</td>
</tr>
<tr>
<td>12-10</td>
<td>10-30</td>
<td>30-33</td>
</tr>
<tr>
<td>21-20</td>
<td>20-30</td>
<td>30-31</td>
</tr>
</tbody>
</table>
Logistic problems: tasks assigned

flight tasks

flight agent

agent 1

loc_{10}

loc_{11}

loc_{12}

loc_{13}

loc_{14}

agent 2

loc_{20}

loc_{21}

loc_{22}

loc_{23}

loc_{24}

agent 3

loc_{30}

loc_{31}

loc_{32}

loc_{33}

loc_{34}

orders as dependent tasks

\begin{align*}
t & < t < t \\
12-10 & < t < 10-20 & 20-24 \\
32-30 & < t < 30-20 & 20-21 \\
12-10 & < t < 10-30 & 30-33 \\
21-20 & < t < 20-30 & 30-31 \\
\end{align*}
Logistic problems: graph representation

Each individual agent plan is a partial ordering of the tasks assigned to it, respecting the existing constraints.
Uncoordinated choice of individual plans

Individual plans are ok.

Joint plan is not feasible!
Distributed partitioning
(Buzing, ter Mors, Valk, Witteveen)

• add a (minimal) set of additional constraints $\Delta C$ to $C$, changing it into another problem that allows complete decomposition;

• solve the planning problems independently;

• compose the plans to a joint plan without making revisions.
Distributed partitioning

- $T^A$ is the set of tasks of agent $A$.
- Dependency information is kept on a common blackboard (or trusted third party).

for round $i = 1, 2, ..., n$

1. each agent $A$ takes a subset $T_i^A$ of tasks from $T^A$ that are not dependent upon tasks of others agents
2. the subset $T_i^A$ is stored and removed from $T^A$
3. after the last round, the agent $A$ has a sequence of nonempty subsets $(T_{i1}^A, T_{i2}^A, T_{i1}^A, ..., T_{ik}^A)$
4. each agent $A$ adds constraints such that $T_{ij}^A < T_{i(j+1)}^A$ for all $j$

until $T^A = \emptyset$
Example: application of partitioning

- Task city 1
- Task city 2
- Task city 3

Flights and tasks not dependent on other tasks in round 1.
Example: application of partitioning

- Task city 1
- Task city 2
- Task city 3

Tasks not dependent on other tasks in round 2
Example: application of partitioning

- Task city 1
  - 1
  - 2
- Task city 2
  - 3
- Task city 3
  - 1
  - 2
  - 3

Flights are indicated by the connections between the task cities. Tasks not dependent on other tasks in round 3 are highlighted in orange.
Example: application of partitioning

Task city 1

Task city 2

Task city 3

Flights

Local dependency constraints added
Example: application of partitioning

With these dependency constraints added, every set of individual plans leads to a coordinated joint plan. Check!
Partitioning strategies

distributed task splitting algorithms

polynomial distributed algorithm splitting agent tasks into ordered partition

coordination protocols (cf specialized social laws)

rule for agents to obtain coordination constraints
Example: protocol for coordination

- **task city 1**
- **task city 3**
- **flights**
- **task city 2**

**Simple coordination protocol**
- **city tasks:**
  - add constraints such that pre-flight + local transportation tasks precede all post-flight tasks

**Flight tasks:**
- no restrictions
Coordination during planning

Coordination and planning as intertwined processes; coordination is applied to partial plans.

Example: (G)PGP, COABS

Agents do not plan autonomously but interact during planning.
Coordination during Planning

method

• agents communicate local plans/goals and coordinate them with other agents
• as a result, local plans may be revised;
• commitment to realization of goals of other agents
• agents enrich their own local view by including plans/goals of others

consequences

• iterative refinement of coordination between single agent plans
• (local) plan generation is interleaved with coordination steps (iteratively)
• sharing of partial views on goals and their dependencies
Coordination during Planning

(G)PGP’s approach (Durfee, Decker, Lesser)

- coordination module for local planners to handle dependencies
- uncoordinated (abstract) plans are given to the coordination module
- the coordination module applies coordination mechanisms (by communicating with other modules) to produce coordinated plans

Example

we use the logistic example to illustrate GPGP
Example: logistic problem

Agents $i$ are responsible for local city transportation tasks.

Flight agent handles airport to airport transportation.

Transportation tasks consist of interdependent city pre-transportation + flight + city post-transportation tasks.

*problem* create coordinated local plans.

---

**Example: logistic problem**

- **Agents**: $i$
- **Tasks**: transportation
- **Areas**: city, airport
- **Roles**: flight agent, agents 1, 2, 3
Task-based representation

tasks 1

flight tasks

tasks 2

tasks 3

dependency relations
Local plan development

tasks 1

flight tasks

tasks 2

tasks 3

initial constraints
developed locally
Flight agent starts to coordinate with 2

agents develop initial schedules:

flight agent talks to agent 2:
conflicts detected
Flight coordinates with 2
tasks 1

flight tasks

tasks 2

flight agent and agent 2 change local constraints;
both update their views: interdependencies known
Flight coordinates with 2 tasks:

- Flight agent talks to agent 3:
  - they detect a conflict between their plans.
Flight coordinates with 3

Flight agent and agent 3 change local constraints; both update their views.
Flight coordinates with 3

flight agent detects a conflict with agent 2 schedule
Flight coordinates again with 2

flight agent detects a conflict with agent 2 schedule: both change plan and starting times
Flight coordinates with 1

flight agent talks with agent 1: conflict between starting times.
Flight coordinates with 1

flight agent talks with agent 1: conflict with starting times. flight agent changes plan and schedule
Conflicts detected with 2 and 3

There are conflicts with both agent 2 and agent 3
Flight coordinates with 2 and 3

There are conflicts with both agent 2 and agent 3: as a result of coordination changes in plans and schedules occur.
Flight coordinates with 1

No more conflicts detected; all plans are locally and globally feasible.
Coordination during planning:  
Summary and Comments

- planning and coordination are interleaved processes
- planning agents communicate (partial) plans
  (and are prepared to revise their current local plan)
- offers possibilities to interleave planning + coordination + execution
- communication overload?
- termination and completeness?
Post-planning coordination

- Task
- Agents
- Task refinement
- Task allocation
- Planning (Agent 1, Agent 2, Agent 3, Agent 4)
- Independence planning systems
- Post-planning coordination
  - Improves quality of agent plans
- Joint plan execution
- Examples: plan integration, plan merging, plan fusion

Exploits positive interactions, removes negative interactions (conflicts)
Plan Merging: essentials

- agents have developed plans autonomously
- they are interested to increase efficiency by removing redundancy/incompatibilities from their plans without affecting goal realizability
- *plan merging* aims at merging compatible actions and/or action sequences such that replacement of an action sequence does not affect goal realizability.
Plan merging: using free resources
(de Weerdt, Tonino, Witteveen)

• merge by identifying free resources (plan products) in (other) plans that can be used to realize goals of an agent

• as a result, some operators can be removed.
Example plan merging

passengers: origins

passengers: goal destinations

taxi’s
Plans developed

Plan taxi 1
- pickup p1; drive to B; deliver p1 at B;
- drive to C; pickup p2; drive to D; deliver p2

Plan taxi 2
- drive to C; drive to D; pickup p3; drive to E; deliver p3

By cooperating, there exist better plans
Plan:
take a passenger from A to B and a passenger from C to D using a taxi at A
Plan representation

Plan 1

Plan 2
Basic idea: use free resources

Plan 1

Plan 2

This is a free resource of agent 2xt
Plan representation: merging

the free resource from the other plan is used
this action becomes obsolete
Plan representation: merging

this action becomes obsolete, too
As a result, we have two new interdependent plans. Agent 1 now is dependent upon agent 2 for achieving the goal of transporting passenger 2 from C to D.
Summary and comments

- plan merging suitable for both competitive and cooperative planners to improve plans by removing incompatibilities and exploiting positive interactions

- the quality of the resulting plan is dependent upon the cooperativeness of the participating agents and the quality of the original plans produced

- plan merging algorithms differ
  - in the choice of the plan objects to be merged,
  - the amount of detail required about the plans to be merged and
  - the any-time character of the merging algorithm applied
References


Mahola!
Tutorial Contents

1. Introduction & taxonomy
2. Single-agent planning techniques for multi-agent planning
3. Details of some approaches to coordinate plans
   1. Pre-planning coordination
   2. Coordination during planning (GPGP)
   3. Post-planning coordination (plan merging)
4. (extra) Coordination of self-interested agents
Self-interested agents

Problem setting

• Planning routes for internet packages along self-interested routers

• Coordinating transport companies in a supply chain

• As a government, set up a market for coordinated medical treatments, or public transport

... mechanism design!
Mechanism design

• sub-field of economic theory from engineering perspective

• assuming agents are rational, find a way to obtain a social choice (joint decision), eg
  • elections
  • markets
  • auctions
  • government policy
Multi-agent planning for self-interested agents ⊂ mechanism design

• as system designers our goal is to optimize global welfare

• while agents may act selfishly...
Some results from game theory

Def: **unanimity**: if every wants A, choose A

Def: **independence of irrelevant alternatives**: if some irrelevant option is removed, the choice should stay the same

Def: **dictatorship**: one agent determines every choice

Arrow’s theorem: every social welfare function over a set of more than 2 options that satisfies unanimity and independence of irrelevant alternatives is a dictatorship.

Clearly, we cannot hope for a good solution to MAP. Or can we?
Mechanisms with money

• Vickrey’s second price auction
Recommended reading