From One to Many: Planning for Loosely Coupled Multi-Agent Systems

Ronen I. Brafman[1] Carmel Domshlak[2]

[1]Ben-Gurion University, Israel

[2]Technion, Israel

ICAPS 2008

Motivation

Logistics planning

Deliver packages using vehicles (trucks, airplanes, ships) operating in/between different countries/regions/cities

- Classical benchmark for "single-agent" planning
- Classic example of a distributed system $\rightsquigarrow$ vehicle = agent

(Informal) Question

Can we exploit the fact that the domain describes a naturally distributed system to make planning more efficient?

(Ultimate) Answer

YES, we can solve distributed components independently

Motivation

Logistics planning

Deliver packages using vehicles (trucks, airplanes, ships) operating in/between different countries/regions/cities

- Classical benchmark for "single-agent" planning
- Classic example of a distributed system $\rightsquigarrow$ vehicle = agent

(Informal) Question

Can we exploit the fact that the domain describes a naturally distributed system to make planning more efficient?

(Ultimate) Answer

YES, we can solve distributed components independently

Basic Motivation/Intuition

k-agents MA Systems (Logistics domain example)

Fully decoupled

Vehicles are a priori responsible for different packages

Same as planning k times for a single agent
$\leadsto$ linear time-complexity growth
($\exp(k)$ time-complexity reduction)

Fully coupled

Vehicles have to move the same packages and maybe coordinate on loads/unloads

Same as planning for a single "k-times larger" agent
$\leadsto$ $\exp(k)$ time-complexity growth
(no reduction in time-complexity)

Basic Motivation/Intuition

k-agents MA Systems (Logistics domain example)

Fully decoupled

Same as planning k times for a single agent
$\rightsquigarrow$ linear time-complexity growth
($\exp(k)$ time-complexity reduction)

Fully coupled

Same as planning for a single "k-times larger" agent
$\rightsquigarrow$ $\exp(k)$ time-complexity growth
(no reduction in time-complexity)

Loosely coupled

Somewhere in between depending on the "level" of coupling?

Available results? *None that we know of*

“Loose Coupling” is a Loose Concept

Questions

1. How to measure the coupling level of a MA system?

2. Is there an algorithm that
   1. can handle any “coupling level”, yet
   2. is guaranteed to benefit from lower “coupling level”

Contributions

- Formal measure of coupling level by a combination of
  1. a measure of a MA system's inherent coupling level
  2. a measure of a problem's coupling level

- An algorithm that scales
  - exponentially with coupling level
  - polynomially with the number of agents

- Based on a very simple model
  $\rightsquigarrow$ a minimal extension of STRIPS to MA systems

Our Focus Here

Input Planning problem for a set of k collaborative agents

Question To what extent is planning for such a MA system harder than solving individual planning problems of each of the agents in isolation?

Approach Theoretical. Try to formulate an algorithm that is tractable under reasonable conditions.

Centralized Planning for MA Systems

Our Focus Here

Input: Planning problem for a set of k collaborative agents

Question: To what extent is planning for such a MA system harder than solving individual planning problems of each of the agents in isolation?

Approach: Theoretical. Try to formulate an algorithm that is tractable under reasonable conditions.

Main Ideas

A New Graphical Model

Potential (positive and negative) interactions between the agents' individual abilities (= actions)

System coupling-level

Define an interaction graph of the system

Nodes = agents

Edges = agents need (to coordination with) each other

Parameter $\omega \rightsquigarrow$ tree-width of interaction graph

Main Ideas

A New Graphical Model

Potential (positive and negative) interactions between the agents' individual abilities (= actions)

System coupling-level

Parameter $\omega \rightsquigarrow$ tree-width of interaction graph

Problem coupling-level

Some problems require more coordination than others!

Parameter $\delta \rightsquigarrow$ minmax number of times a single agent needs some other agent to do something for it

Main Ideas

System coupling-level

Parameter $\omega \rightsquigarrow$ tree-width of interaction graph

Problem coupling-level

Parameter $\delta \rightsquigarrow$ minmax number of times a single agent needs some other agent to do something for it

Algorithm

- Fix the agents' commitments to each other
  $\rightsquigarrow$ *careful selection of language matters!*
- Let each agent independently plan "in-between" commitments
- Use iterative deepening to extend the number of per-agent commitments if needed

Agent Actions

Logistics planning

Deliver packages using vehicles (trucks, airplanes, ships) operating in/between different countries/regions/cities

- Actions `move`$(v, from, to)$, `load`(p, v, at), `unload`(p, v, at)
- Agents: vehicles
- Vehicle agent actions:
  moving it, loading into it, unloading from it

From STRIPS to MA-STRIPS

Everything is the same, except that
actions are partitioned between the agents

Solving MA-STRIPS Problems

Standard Approaches

1. Compile into a single-agent STRIPS problem
   - ☹ Lose all structure and obtain k-times larger "agent"
   - ☹ Worst-case complexity exponential in description size or shortest plan (depending on search strategy)
2. Try to solve as much as possible locally and compose the resulting individual agent plans
   - ☹ What can we say about it?

Solving MA-STRIPS Problems

Standard Approaches

1. Compile into a single-agent STRIPS problem
   - ☹ Lose all structure and obtain k-times larger "agent"
   - ☹ Worst-case complexity exponential in description size or shortest plan (depending on search strategy)
2. Try to solve as much as possible locally and compose the resulting individual agent plans
   - ☹ What can we say about it?

A Closer Look at Agent Actions

Private vs. Non-Private

Private affect and depend only on that agent

Non-Private all the rest

Logistic planning

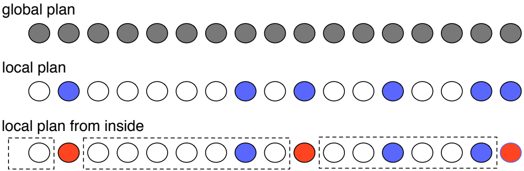
- Move actions are private
  (influence and influenced only by vehicle location)
- Loading into/unloading from a vehicle is non-private
  $\leadsto$ except if the package location is private to the vehicle!

A Closer Look at Agent Subplans

Private vs. Non-Private

Private affect and depend only on that agent

Non-Private all the rest



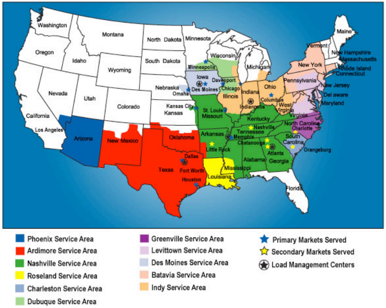
- non-private actions in the plan $\rightsquigarrow$ coordination points
- arbitrarily long sequences of private actions between adjacent non-private actions

Example: Logistics

Logistics

- imagine vehicles moving on a large map
- each vehicle has a service region

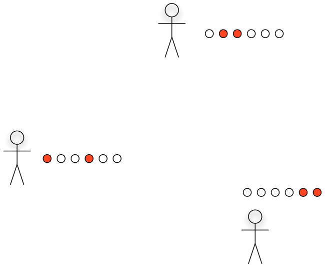
⇝ between each load/unload action, there are multiple `move` actions by the vehicle



Main Idea

"Algorithm"

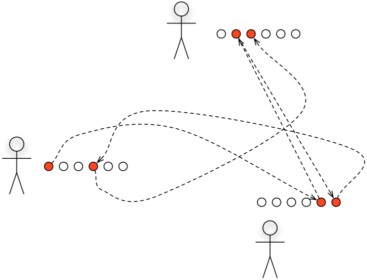
1. Find a good choice of coordination points
2. Solve k local planning problems over the private actions of the agents only



Main Idea

"Algorithm"

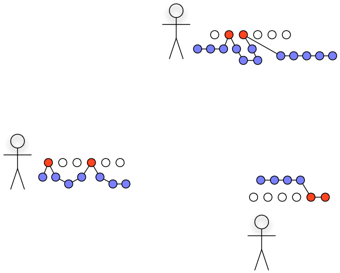
1. Find a good choice of coordination points
2. Solve k local planning problems over the private actions of the agents only



Main Idea

"Algorithm"

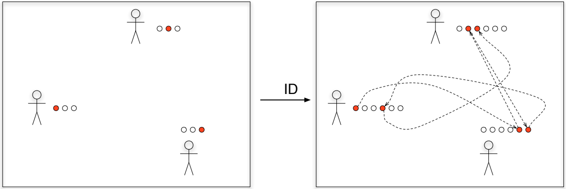
1. Find a good choice of coordination points
2. Solve k local planning problems over the private actions of the agents only



Main Idea

"Algorithm"

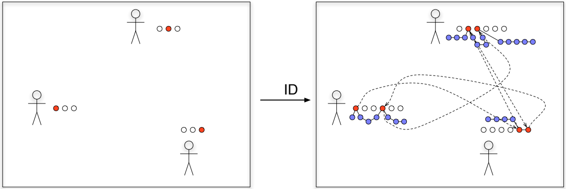
1. Find a good choice of coordination points
    - Iterative deepening on δ — # of coord-points per agent
    - For each choice of δ
        - Define a CSP whose solutions are consistent assignments to the coordination points
2. Solve k local planning problems over the private actions



Main Idea

"Algorithm"

1. Find a good choice of coordination points
2. Solve k local planning problems over the private actions
   - purely independent phase $\rightsquigarrow$ unary constraints
   - can be reduced to regular STRIPS planning



Complexity

The complexity is derived from

1. number of agents (k)
2. complexity of local planning (single)
3. number of "coordination" CSPs we have to solve ($\rightsquigarrow \delta$)
4. solving each "coordination" CSP (?)

Complexity

The complexity is derived from

1. number of agents (k)
2. complexity of local planning (single)
3. number of "coordination" CSPs we have to solve ($\rightsquigarrow \delta$)
4. solving each "coordination" CSP

$$O\left(k \cdot \left(\exp\left(\omega\delta + \omega + \delta\right) + \text{single} \cdot \exp\left(\delta\right)\right)\right)$$

Summary

- Formal measure of coupling level by a combination of
  - δ problem-specific #times an agent needs assistance
  - ω the inherent coupling level of the system
- Planning complexity polynomial in the number of agents (for fixed coupling level)
- "Coordination complexity" is not affected by the length of the local plans
- Generating fully distributed algorithm conceptually easy
  - Use distributed CSP
  - Local planning is already distributed

From STRIPS to MA-STRIPS

Definition

A STRIPS problem is given by a quadruple $\Pi = \langle P, A, I, G \rangle$, where:

- P is a finite set of *atoms*, $I \subseteq P$ is the *initial situation*, and $G \subseteq P$ encodes the *goal* situations,
- Each action $a \in A$ is given by $\langle \mathsf{pre}(a), \mathsf{add}(a), \mathsf{del}(a) \rangle$.

From STRIPS to MA-STRIPS

Definition

An MA-STRIPS problem for a system of agents $\Phi = \{\varphi_i\}_{i=1}^{k}$ is given by a quadruple $\Pi = \langle P, \{A_i\}_{i=1}^{k}, I, G\rangle$, where:

- P is a finite set of *atoms*, $I \subseteq P$ is the *initial situation*, and $G \subseteq P$ encodes the *goal* situations,
- For $1 \leq 1 \leq k$, A_i is the set of actions that the agent φ_i is capable of performing. Each action $a \in A = \bigcup A_i$ is given by $\langle \mathsf{pre}(a), \mathsf{add}(a), \mathsf{del}(a)\rangle$.