Multiagent Systems

7. Working Together: Methods for Coordination

B. Nebel, C. Becker-Asano, S. Wölfl

Albert-Ludwigs-Universität Freiburg

June 4, 2014

7.1 Motivation

Cooperative Distributed Problem Solving (CDPS)

Basic question:

- How can a loosely coupled network of problem solvers work together to solve problems that are beyond their individual capabilities? (Durfee et al., 1989, after Wooldridge, 2009)

Case one:
Agents are **benevolent**, share a common goal and no potential of conflict between them.
⇒ greatly simplifies designer's task

Case two:
Societies of **self-interested** agents don't share a common goal.
⇒ designed by different individuals or organizations to reach goals through cooperation (which is nevertheless inevitable)

Second case is more challenging and interesting!

CDPS versus parallel problem solving

How is CDPS different from *parallel problem solving* (PPS)?

- PPS exploits parallelism to solve problems
- Computational components are processors, not agents
- Central node **decomposes** overall problem into subcomponents
- Processors independently provide subsolutions
- Subsolutions are **assembled** into overall solution by central node

PPS was synonym for CDPS in "early days of multiagent systems", two fields are now quite separate.

How to evaluate a MAS

What criteria can be used to evaluate the "success" of a multiagent system?

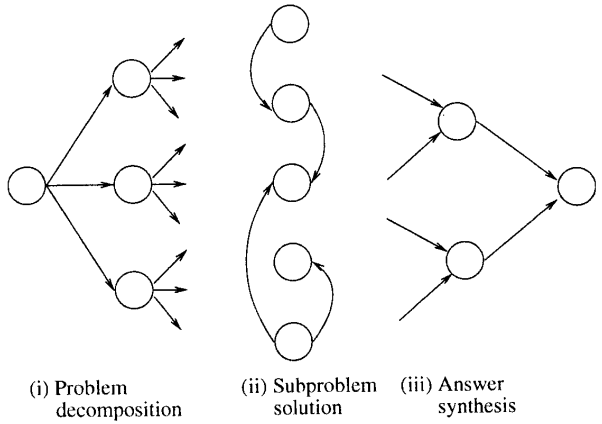
- **Coherence**: How well does the system "behave as a unit"?
  Criteria: solution quality, efficiency of resource usage, conceptual clarity of operation, performance in the presence of uncertainty/failure
- **Coordination**: the degree to which agents can avoid ineffectual activities
  Criteria: amount of communication, task sharing, result sharing

Main issues in CDPS

Main issues to be addressed in CDPS include:

- How can a problem be **divided into smaller tasks** for distribution among agents?
- How can problem solution be **effectively synthesized** from subproblem results?
- How to **optimize overall problem-solving** activities of agents so as to produce solution to maximize **coherence** metric?
- What techniques can be used to **coordinate activity** of agents, thus avoiding destructive interactions, and maximizing effectiveness?

The three stages of CDPS



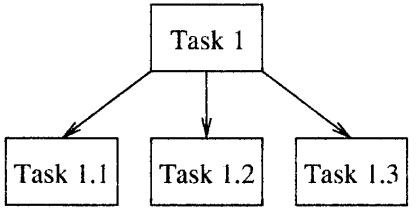
(Wooldridge, 2009, p. 154)

7.2 Task sharing

- Contract Net protocol
- Blackboard Architecture

Task sharing

"How can a problem be divided into smaller tasks for distribution among agents?"
⇒ Task-sharing



(Wooldridge, 2009, p. 156)
A task is decomposed into subproblems that are allocated to agents.

Task sharing in the Contract Net

The Contract Net (CNET) protocol (Smith 1977, 1980):

- high-level protocol for achieving efficient cooperation through task sharing
- basic metaphor: contracting

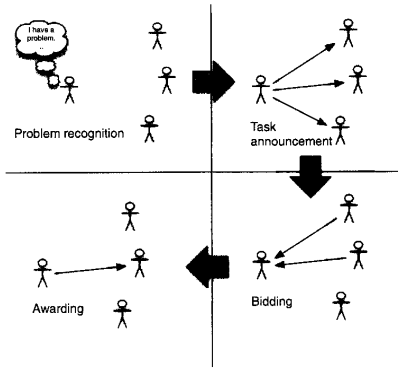
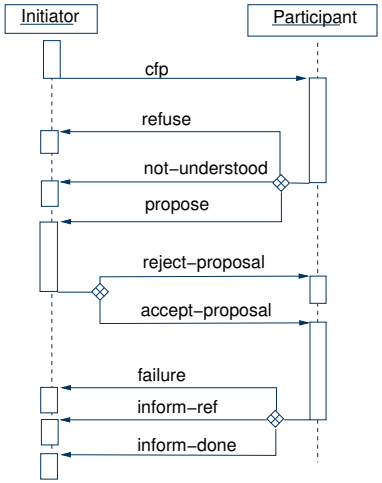


Figure 8.3: The Contract Net (CNET) protocol for task allocation.

Contract Net protocol

- One of the **oldest, most widely used** agent interaction protocols
- A manager agent announces one or several tasks, agents place bids for performing them
- Task is assigned by **manager** according to evaluation function applied to agents' bids (e.g., choose cheapest agent)
- Idea of **exploiting local cost function** (agents' private knowledge) for distributed optimal task allocation
- Even in purely cooperative settings, **decentralization** can **improve global performance**
- A typical example of "how it can make sense to agentify a system"
- Successfully applied to different domains (e.g. transport logistics)

Contract-net protocol



(FIPA Contract Net Interaction Protocol Specification, 2000)

Marginal cost of task

How should a potential contractor decide whether or not to bid for a task?
Idea: marginal cost for task (Sandholm 1999)

- Let t be the time, i be the contractor, τ_i^t be the task scheduled for i at time t
- The *endowment* of contractor i is e_i and an announcement ts is made with $\tau(ts)$ denoting the task specified by ts
- Let $c_i^t(\tau)$ denote the *cost* to agent i of carrying out the set of task τ at time t

Then, the **marginal cost** of carrying out tasks τ is denoted by:

$$\mu_i(\tau(ts) \mid \tau_i^t) = c_i(\tau(ts) \cup \tau_i^t) - c_i(\tau_i^t)$$

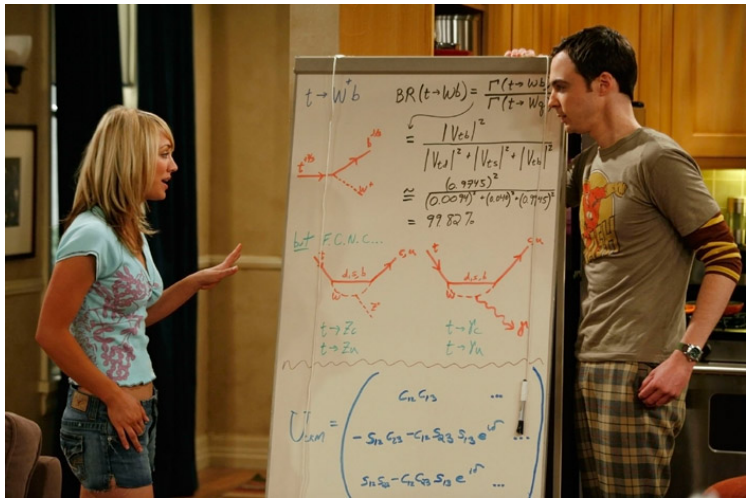
$\Rightarrow$ difference between cost of carrying out old plus new tasks and only carrying out previously agreed tasks

Blackboard architecture

Inspired by human problem solving in a team:

> "Imagine a group of human specialists seated next to a large blackboard. The specialists are working cooperatively to solve a problem, using the blackboard as the workplace for developing the solution.
> Problem solving begins when the problem and initial data are written onto the blackboard. The specialists watch the blackboard, looking for an opportunity to apply their expertise to the developing solution. When a specialist finds sufficient information to make a contribution, she records the contribution on the blackboard, hopefully enabling other specialists to apply their expertise. This process of adding contributions to the blackboard continues until the problem has been solved."
> (Corkill, 1991)

Blackboard metaphor



Blackboard architecture characteristics

Eight characteristics of blackboard architectures (Corkill, 1991):

- **Independence of expertise** (I think, therefore I am.)
- **Diversity in problem-solving techniques** (I don't think like you do.)
- **Flexible representation of blackboard information** (If you can draw it, I can use it.)
- **Common interaction language** (What'd you say?)
- **Positioning metrics** (You could look it up.)
- **Event-based activation** (Is anybody there?)
- **Need for control** (It's my turn.)
- **Incremental solution generation** (Step by step...)

Basic Blackboard System

A basic blackboard system (Corkill, 1991):

- KS: static Knowledge Source
- KS Activations: Combination of KS knowledge and specific "triggering context" ⇒ the proactive "agent"

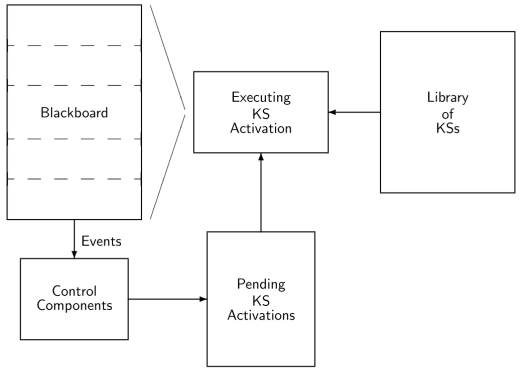


Figure 2: Basic Blackboard System

Blackboard architecture components

The simple blackboard system consists of two major components:

1. The **blackboard** as global structure available to all KSs, serves as:
   - a community memory of raw input data; partial solutions, alternatives, and final solutions; and control information
   - a communication medium and buffer
   - a KS trigger mechanism
2. **Control component** as explicit control mechanism, chooses course of action based on state of blackboard and set of triggered KSs

Incremental reasoning style with the solution being built one step at a time. At each step:

- execute any triggered KS
- choose a different focus of attention, on the basis of the state of the solution

When to use Blackboard Problem-Solving Approach

Situations suitable for the blackboard approach (Corkill, 1991):

- diverse, **specialized knowledge** representations
- as an **integration framework** for heterogeneous problem-solving representations and expertise (e.g., diagnostic systems)
- software development in teams of independent developers (i.e. as **software-engineering** metaphor)
- When "**uncertain knowledge** or limited data inhibits absolute determination of a solution"
- To realize **multilevel reasoning** or flexible, dynamic control of problem-solving activities

The "Cognitive Agent Architecture – **Cougaar**" (DARPA) is based on a blackboard architecture, but not FIPA-compliant. It is used for the development of applications in military logistics.

7.3 Coordination

Methods for coordination

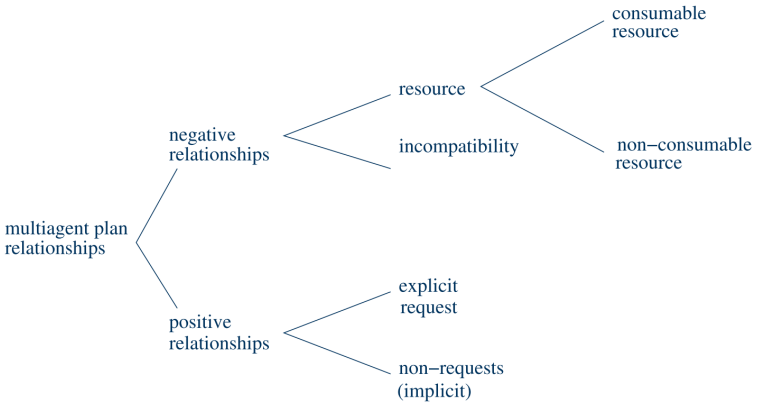
Coordination

Coordination is the process of managing inter-dependencies between agents' activities

- Coordination is a special case of interaction in which agents are aware how they depend on other agents and attempt to adjust their actions appropriately.
- Actually this only covers agent-based coordination, but there can also be centralised mechanisms
- In contrast to cooperation, coordination is also necessary in non-cooperative systems (unless agents ignore each other)

Coordination relationship typology

A typology in the context of multiagent planning is provided by (von Martial, 1990):



von Martial's typology

von Martial's typology distinguishes:

- Positive relationships: relationships between two agents' plans for which benefit will be derived for at least one agent if plans are combined
- Requests: explicitly asking for help with own activities
- Non-requested: pareto-like implicit relationships
  - action equality relationships: sufficient if one agent performs action both agents need
  - consequence relationships: side effects of agent's plan achieve other's goals
  - favor relationships: side effects of agent's plan make goal achievement for other agent easier

Basic difference to traditional computer systems
⇒ coordination is achieved **at run time** rather than **design time**

7.4 Models for Coordination

- Partial global planning
- Joint intentions
- Mutual modeling
- Thanks

Partial global planning

Partial global planning (PGP): exchange information to reach common conclusions about problem-solving process

- Partial ⇒ individual agents don't generate plan for entire problem
- Global ⇒ agents use information obtained from others to achieve non-local view of problem
- Three iterated stages:
  1. Agents deliberate locally and generate short-term plans for goal achievement
  2. They exchange information to determine where plans and goals interact
  3. Agents alter local plans to better coordinate their activities
- **Meta-level structure** guides the coordination process, dictates information exchange activities

Partial global planning

Central data structure is the **partial global plan** containing:

- Objective: larger goal of the system
- Activity maps: describe what agents are doing and the results of these activities
- Solution construction graph: describes how agents should interact and exchange information to achieve larger goal

PGP extension

The Partial Global Planning framework was extended to the **Generalized PGP** (GPGP), which introduces five techniques for coordinating activities, i.e. strategies for:

- updating non-local viewpoints (share all/no/some inform.)
- communicating results
- handling simple (action) redundancy
- handling hard ("negative") coordination relationships (mainly by means of rescheduling)
- handling soft ("positive") coordination relationships (rescheduling whenever possible, but not "mission critical")

Joint intentions

Discussed intentions in practical (single-agent) reasoning:

- Intentions also provide **stability** and **predictability** necessary for social interaction
  $\Rightarrow$ intentions also significant for coordination, especially teamwork
- Helps to **distinguish between non-cooperative and cooperative** coordinated activity

Basic question:
In which way are individual intentions **different** from (and what role do they play in) collective intentions?

Remember Cohen and Levesque's theory of intentions?
They extended it to teamwork situations, introducing a notion of "responsibility"

Joint intentions: example

- Imagine: We try to lift a stone together, and I discover it won't work
  $\Rightarrow$ individually rational behavior: drop the stone!
- However, this is not really cooperative (we should at least inform other)
- Two important notions:
  1. **commitments** (pledges or promises to underpin an intention)
  2. **conventions** (mechanisms for monitoring commitment, mechanics of adopting/abandoning commitments)
- Agents commit themselves to actions or states of affairs
- Commitments are persistent, i.e. they are not dropped unless special circumstances arise
- Conventions define these circumstances, e.g., motivation for goal is no longer present, or it is / can never be achieved

Teamwork-based model of CDPS

Teamwork-based model of CDPS (Wooldridge & Jennings, 1994, 1999): a practical model of how CDPS can operate using a teamwork approach

- Stage 1: **Recognition of a goal** that can be achieved through cooperation (e.g. an agent can't do it (efficiently) on his own)
- Stage 2: **Team formation**, i.e. assistance solicitation
  - if successful, this results in nominal commitment to collective action
  - deliberation phase, ends in agreement on ends (not on means)
  - rationality plays a role in deciding whether to form a group
- Stage 3: **Plan formation** involves joint means-ends reasoning, e.g. through negotiation or argumentation
- Stage 4: **Team action** plan of joint action is executed by agents based on mutual convention

Mutual modeling

Basic idea: Putting oneself in the shoes of the other.

- Involves modeling others' beliefs, desires, and intentions
- ...and coordinating own actions depending on resulting predictions
- Explicit communication unnecessary
- MACE one of the first systems (mid 1980s) to use **acquaintance models** for this purpose
- Acquaintance knowledge involves information about others
  - **Name** unique to every agent
  - **Class** of group the agent belongs to
  - **Roles** played by an agent in a class
  - **Skills** as the capabilities of the modeled agent
  - **Goals** that the modeled agent wants to achieve
  - **Plans** as the agent's means to achieve an end
- Agent explicitly models itself!

Norms and social laws

Norms are established patterns of expected behavior, **social laws** often add some authority to that (can be enforced or not)
⇒ Balance autonomy with goals of entire society

Features of norms and social laws in terms of conventions for MAS:

- conventions make decision making **easier** for agents
- conventions can be designed **offline** (simpler) or **emerge** from within the system (more flexible)
- Hard to predict at design time which norm will be **optimal**
- Also hard to derive **global conventions** from agents' local point of view

Emergent social norms and laws I

Example: the t-shirt game

- agents wear red or blue t-shirt (initially at random), goal is for everyone to wear the same color
- agents are randomly paired in each round of the game, get to see other's t-shirt color, and then may decide to switch color

Problem: agent must decide which convention to adopt although no global information is available
Possible update functions (=decision rules based on history):

- **Simple majority**: agent chooses color observed most often
- **Simple majority with agent types**: agents confide in certain other agents and exchange memory with them to inform their decision
- **Simple majority with communication on success**: agents will communicate (successful part of) memory if success rate exceeds a threshold
- **Highest cumulative reward**: uses strategy that has had the highest cumulative reward so far

Emergent social norms and laws II

Properties of update functions:

- All update functions converged to some convention
- Measure: time taken to converge
- Memory restarts were investigated to model "new ideas"
- But also stability important (we don't want society to change conventions all the time)
- Basic result: for **highest cumulative reward** rule, for any $0 \leq \epsilon \leq 1$ agents will reach agreement within n rounds with probability $1 - \epsilon$
- Also, once reached, the convention will be stable
- Convention is efficient, i.e. it guarantees payoff no worse than that obtainable from sticking to initial choice
- Note that change of norm may be expensive in practice!

Offline design

Offline design is closely related to mechanism design
Formally, remembering our agent model $Ag : \mathcal{R}^E \rightarrow Ac$ we can define **constraints** $\langle E', \alpha \rangle$ where:

- $E' \subseteq E$, a constraint on the environment
- $\alpha \in Ac$, actions α as before

A **social law** is a **set of such constraints** ⇒ agents/plans are legal if they never attempt to perform forbidden actions

useful social law problem

Given a set $F \subseteq E$ of **focal states** (states that should always be allowed), a "useful social law problem" is to find a social law that allows agents to **legally** visit any state in F.

General problem **NP-complete**, tractable cases "do not appear to correspond to useful real-world cases."

Summary

- Cooperative Distributed Problem Solving (CDPS)
- Coherence & Coordination
- Task sharing: Contract Net & Blackboard
- Coordination
- Partial global planning: PGP & GPGP
- Joint intention: commitments & conventions
- Mutual modeling: Norms, social laws, & conventions

$\Rightarrow$ next time: Multiagent Interactions

Acknowledgments

These lecture slides are based on the following resources:

- Dr. Michael Rovatsos, The University of Edinburgh `http://www.inf.ed.ac.uk/teaching/courses/abs/abs-timetable.html`
- Michael Wooldridge: **An Introduction to MultiAgent Systems**, John Wiley & Sons, 2nd edition 2009.
- Corkill, Daniel D. "Blackboard systems." AI expert 6.9 (1991): 40-47.