Escola Tècnica Superior d'Enginyeria
Universitat Rovira i Virgili

[DΣIM]

Lecture 6: Cooperation in MAS (I) – PGP & GPGP

Multi-Agent Systems

Universitat Rovira i Virgili

MESIIA – Master's Degree in Computer Security Engineering and Artificial Intelligence
MAI – Master's Degree in Artificial Intelligence

Outline

- Coordination
- Kinds of cooperation in MAS
  - *Emergent* cooperation
  - Cooperation with *explicit* communication
    - Deliberative
      - Partial Global Planning
      - Coalition formation
    - Negotiation
      - Contract Net
      - Auctions
      - Voting

1. What is coordination?

"Coordination is the process by which an agent reasons about its local actions and the (anticipated) actions of others to try and ensure that the community acts in a coherent manner"

N. Jennings, 1996

When to use coordination?

- An agent has a choice in its actions within some task, and the choice affects its performance and that of other agents
- The order or time in which actions are carried out affects performance
- When there are sub-problem dependencies, like:
  - It may be impossible to completely solve one sub-problem without first solving (fully or partially) another sub-problem
  - Solving (fully or partially) one sub-problem may make it easier to solve another sub-problem
- Knowing the solution to one sub-problem may obviate the need to solve another

Sub-problem Interdependencies

- Sub-problems are the same/overlapping, but different agents have either alternative methods or data that can be used to generate a solution
- Sub-problems are part of a larger problem in which a solution to the larger problem requires that certain constraints exist among the solutions to its sub-problems

How an agent orders its sub-goals to do and when it communicates the (partial) results of solving a sub-goal can significantly affect global performance

Components of coordination

- Deciding for each agent in the context of a Multi-Agent System
  - What activities it should do and when it should do them What it should communicate, when it should communicate and to whom
- These two issues are intimately connected
- Agents need *Domain* information and *Control* information
- This is a highly complex computational problem, especially if optimal solutions are required

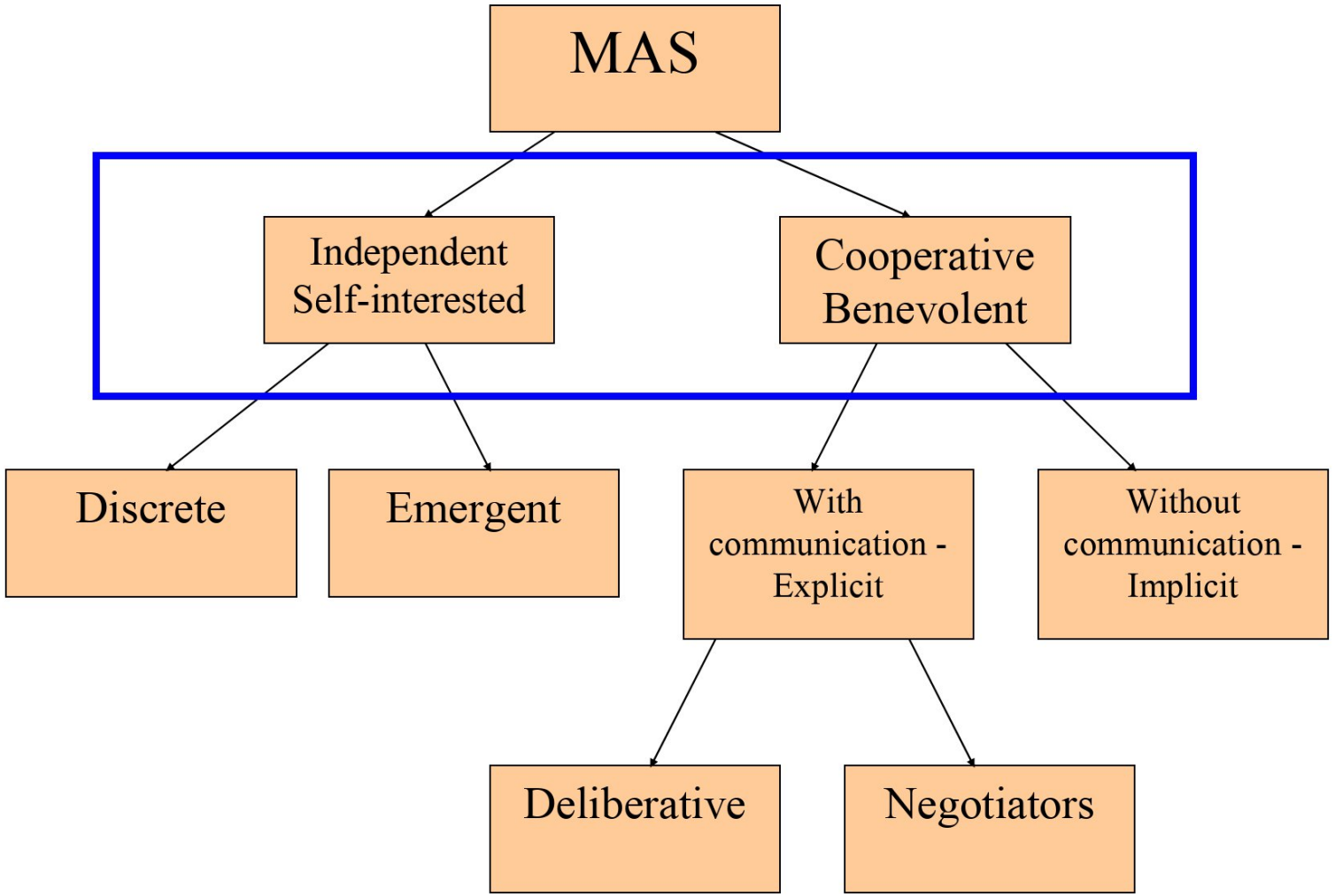
Many kinds of coordination

- Type of information available about *static* and *dynamic* behaviour of agents
  - Cost of acquiring current state of other agents
  - Cost of finding out the abilities of other agents
- Importance of optimal solution
  - Cost of computing coordination decisions
  - Implications of generating non-optimal coordination
- Real-time requirements
  - How long you have to make a decision

✓ *There is no one best approach to Coordination*

✓ *Coordination depends on ability to cooperate*

Cooperation hierarchy (Franklin)



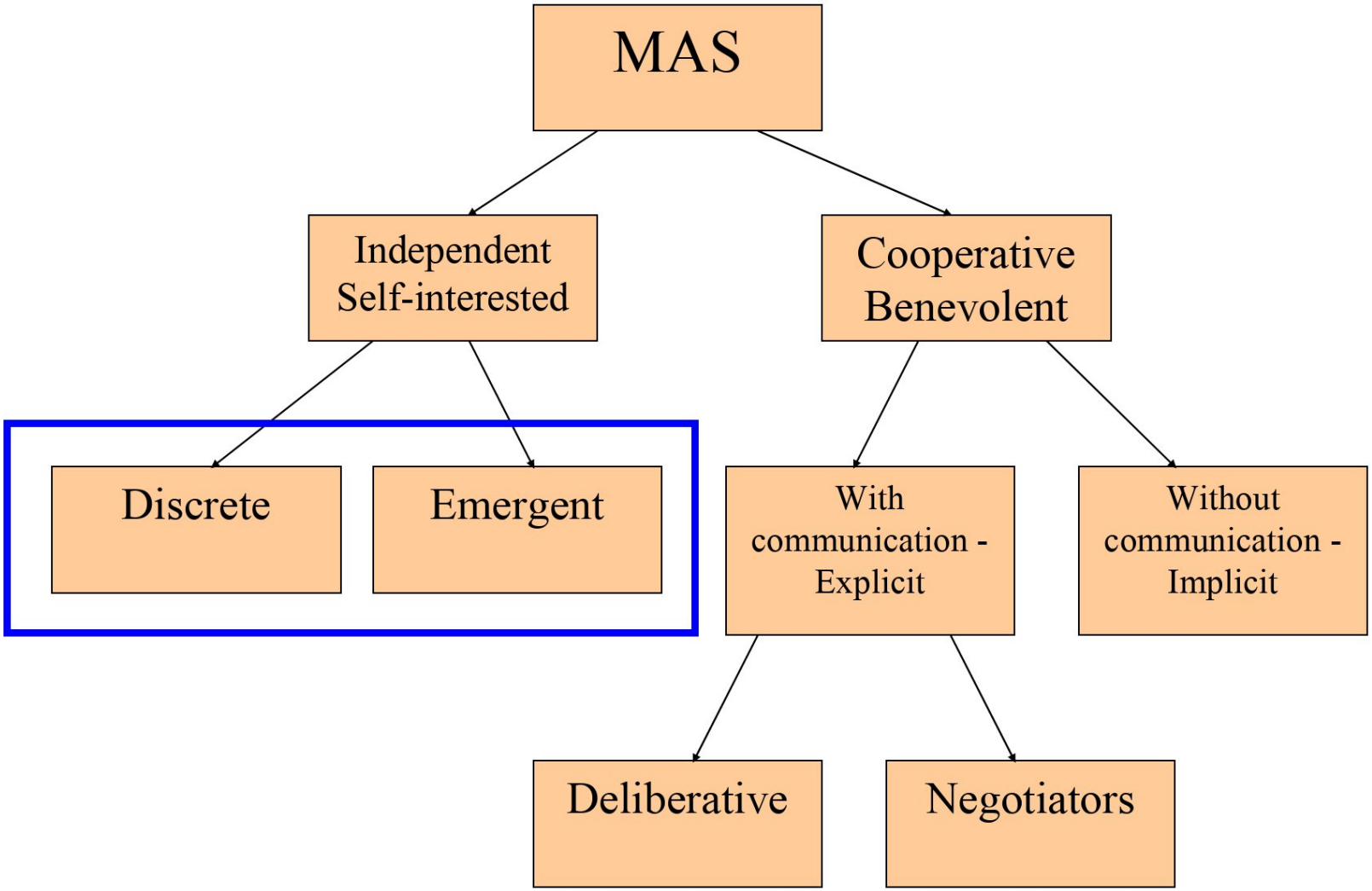
Benevolent Agents

- If we “own” the whole system, we can design agents to help each other whenever asked
- In this case, we can assume agents are *benevolent*: our best interest is their best interest
- Problem-solving in benevolent systems is *cooperative distributed problem solving*
- *Benevolence simplifies the system design task enormously!*

Self-Interested Agents

- If agents represent individuals or organizations, then we cannot make the benevolence assumption
  - E.g. buyers/sellers in e-commerce
- Agents will be assumed to act to further their own interests, possibly at the expense of others
- Potential for *conflict*
- *It may complicate the design task enormously!*

Cooperation hierarchy (Franklin)



Discrete MAS

- Independent agents
- Each agent pursues its own agenda
- The agendas of the agents bear no relation to one another
  - For example, one agent can filter e-mail while another one gathers information from the Web
- No cooperation

MAS with emergent behaviour

- Agents can cooperate with no intention of doing so
- The system can exhibit high-level, complex, intelligent, coordinated behaviour without any designed coordination mechanisms, just as a side effect of the interactions among agents
- Recall *reactive architectures*, and the Luc Steels robots for *Mars exploration*

Example: Stone-gathering robots

- World with a set of stones
- Set of autonomous, independent robots, that can pick up and drop stones
- A simple rule-based individual behaviour of each robot, without any communication or coordination with the other robots, can lead to a complex global system behaviour (putting all the stones together in a pile)

Example: Stone-gathering robots – Behavioural rules (I)

- ***Rule 1 – Pick up stones***

  If (there is not a stone in the gripper) & (there is a stone ahead) ***then*** take the stone in the gripper

- ***Rule 2 – Drop stones together***

  If (there is one stone in the gripper) & (there is a stone ahead) ***then*** drop the stone, go backward for one second and turn at a random angle

Example: Stone-gathering robots – Behavioural rules (II)

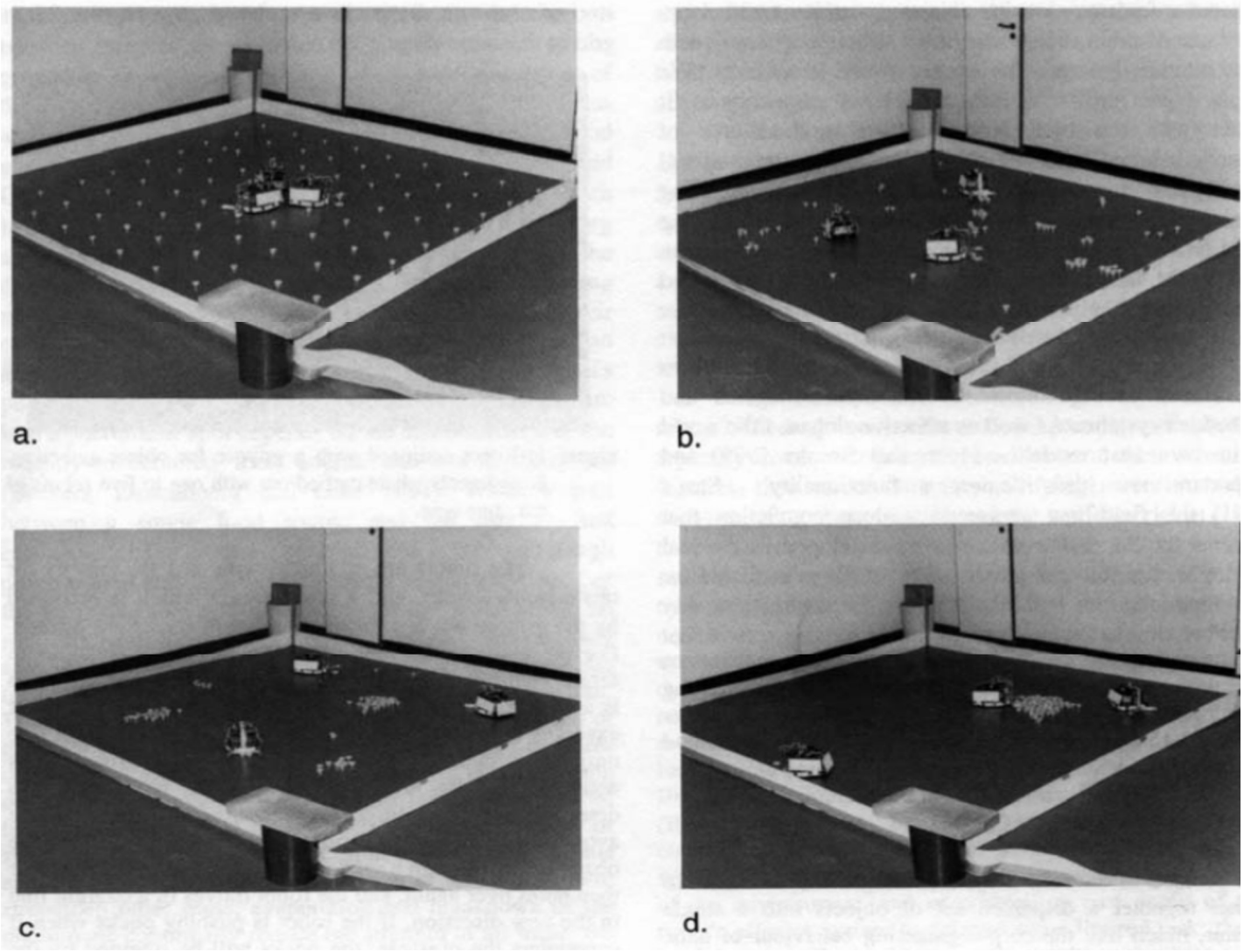
- ***Rule 3 – Exploration***

  If there are no stones ahead ***then*** go forward

- ***Rule 4 – Avoid obstacles***

  If there is an obstacle (wall or other robot) ahead ***then*** avoid the obstacle **(turn** at a random angle and go forward)

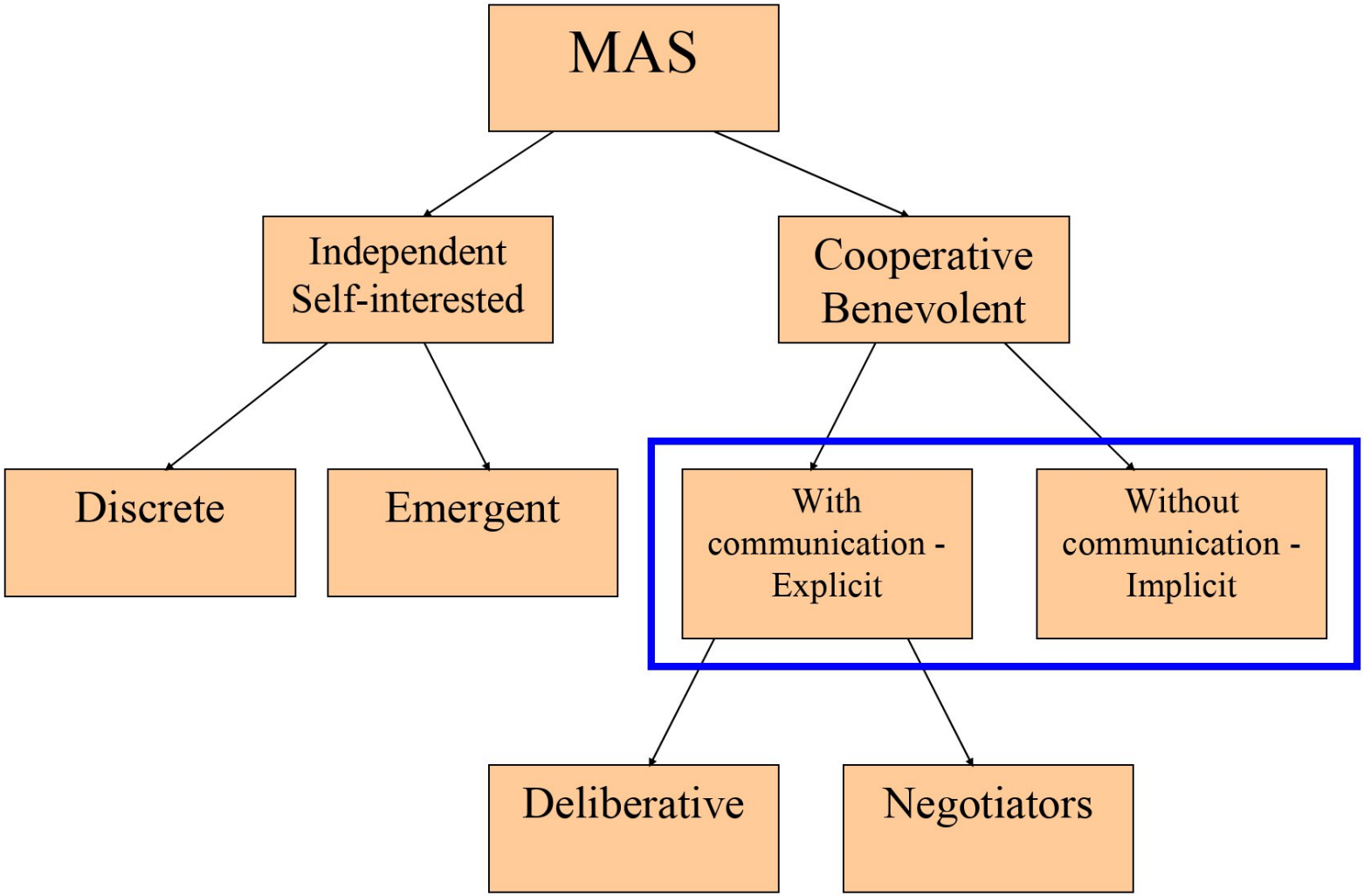
Example: Stone-gathering robots



Important Factors

- Agents
  - Don't need to know about each other, don’t communicate with each other
  - Don't have special roles
  - Losing a few doesn't matter
- Environment
  - Acts as a communication mechanism
  - Is affected by the actions of all individuals
- Result: surprisingly coherent group (coordinated) action
- Natural Intelligence as emergent behaviour

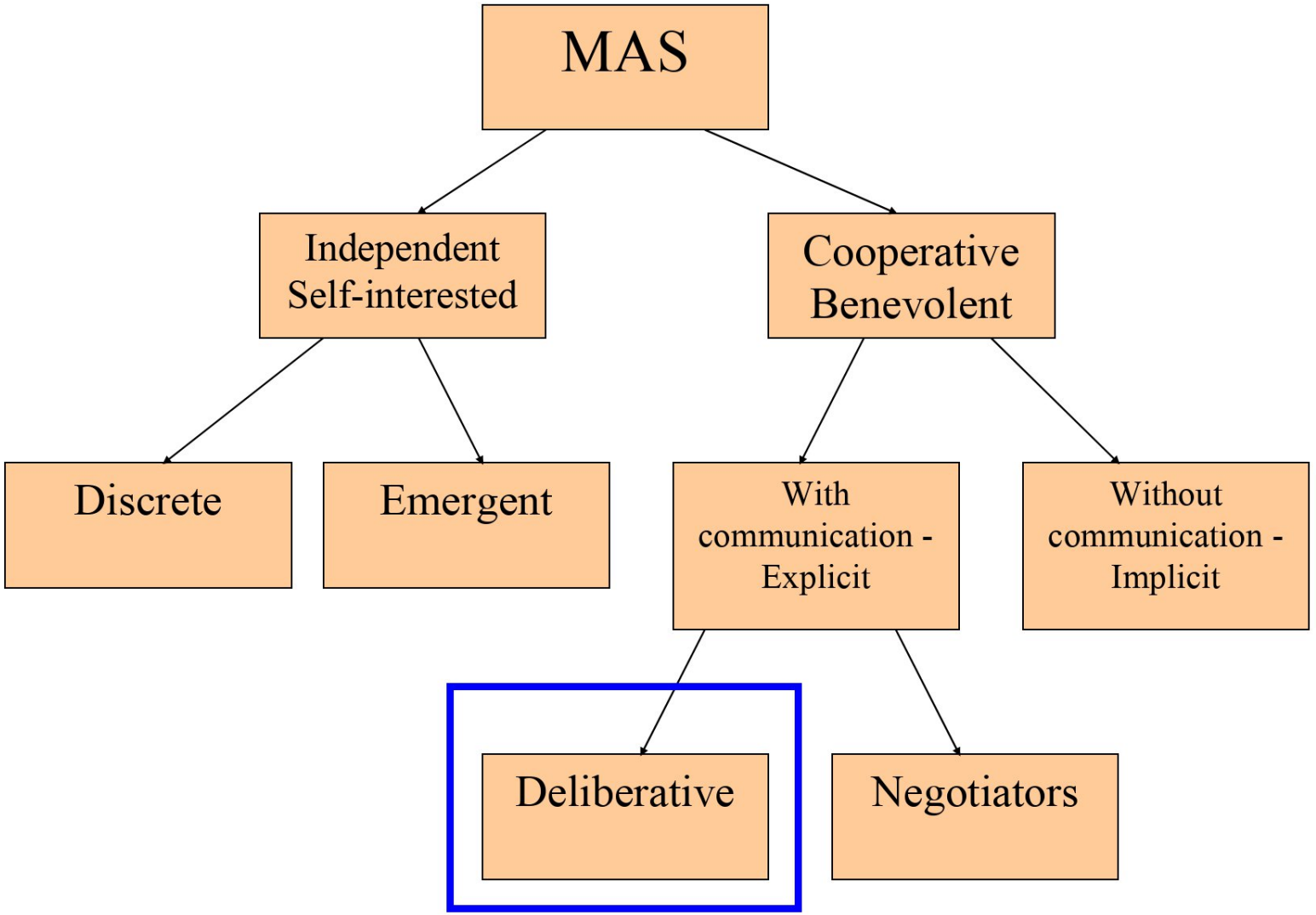
Cooperation hierarchy (Franklin)



Cooperative agents

- The agendas of the agents include cooperating with other agents in the system in some way
  - Explicitly: intentional sending and receiving of communicative signals (e.g., via a common blackboard or via messages)
- Implicitly: without explicit messages (e.g. observing and reacting to the behaviour of the other agents of the system)

Cooperation hierarchy (Franklin)



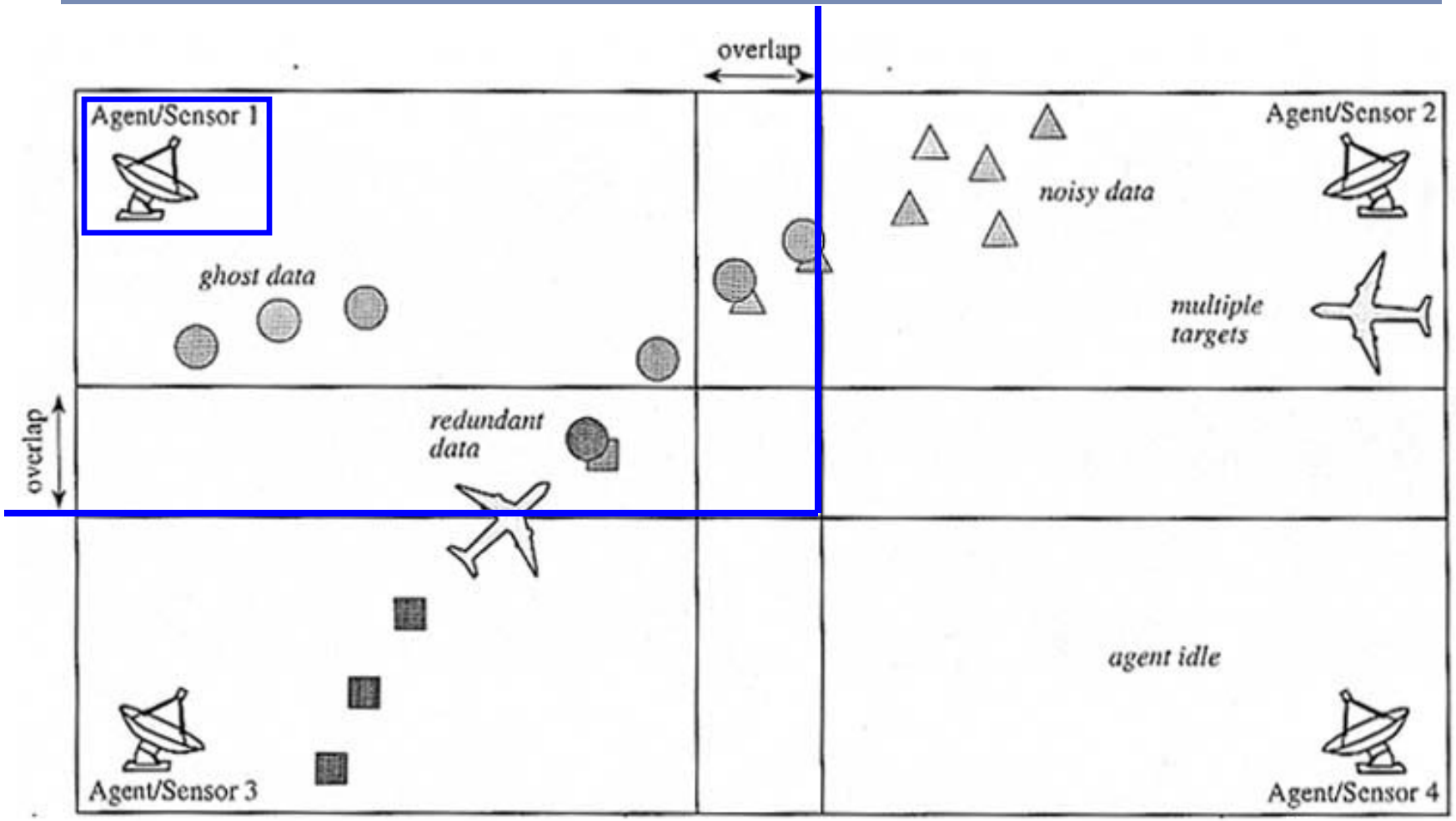
Deliberative agents

- Agents with inference and planning capabilities
- Some kind of explicit distributed organization mechanism, based on *information exchange*, addressed to solve collectively a given problem
- In this course:
  - Partial Global Planning
  - Coalition Formation

Partial Global Planning (PGP) – Durfee, Lesser

- Distributed planning technique
- Integrates planning and execution
  - Not the usual Planning-Scheduling-Action cycle
- The tasks are inherently distributed and each agent performs its own tasks
- Initially applied in the Distributed Vehicle Monitoring (DVM) problem, then extended to be domain independent: Generalised Partial Global Planning

Distributed Vehicle Monitoring (DVM)



Goal in the DVM problem

- The agents are not aware of the global state of the system; however, there is a common goal: *converge on a consistent map of vehicle movements by integrating the partial tracks formed by different agents into a single complete map or into a consistent set of local maps distributed among agents*
- Coordination by means of *partial plans exchange*

Difficulties in DVM (I)

- The data sensed in an area cannot be exhaustively processed in a timely manner
  - Huge volume of incoming data
- Many noisy data generated by sensors/environment, which should not be processed
- Correlations between data sensed in nearby locations provide constraints on whether/how that data should be processed

Difficulties in DVM (II)

- Sensor overlap implies possible processing duplication
  - The same data should not be processed by different agents
- Sensing demands in an area vary heavily dynamically
  - The allocation of work should be done in a very dynamic way, depending on the workload of each agent at each moment

Partial Global Planning phases

1. Create local plans
2. Exchange local plans
3. Generate Partial Global Plans
4. Optimize Partial Global Plans

PGP: Create local plans

1. Create the local plan of each agent

   A plan is a sequence of actions.

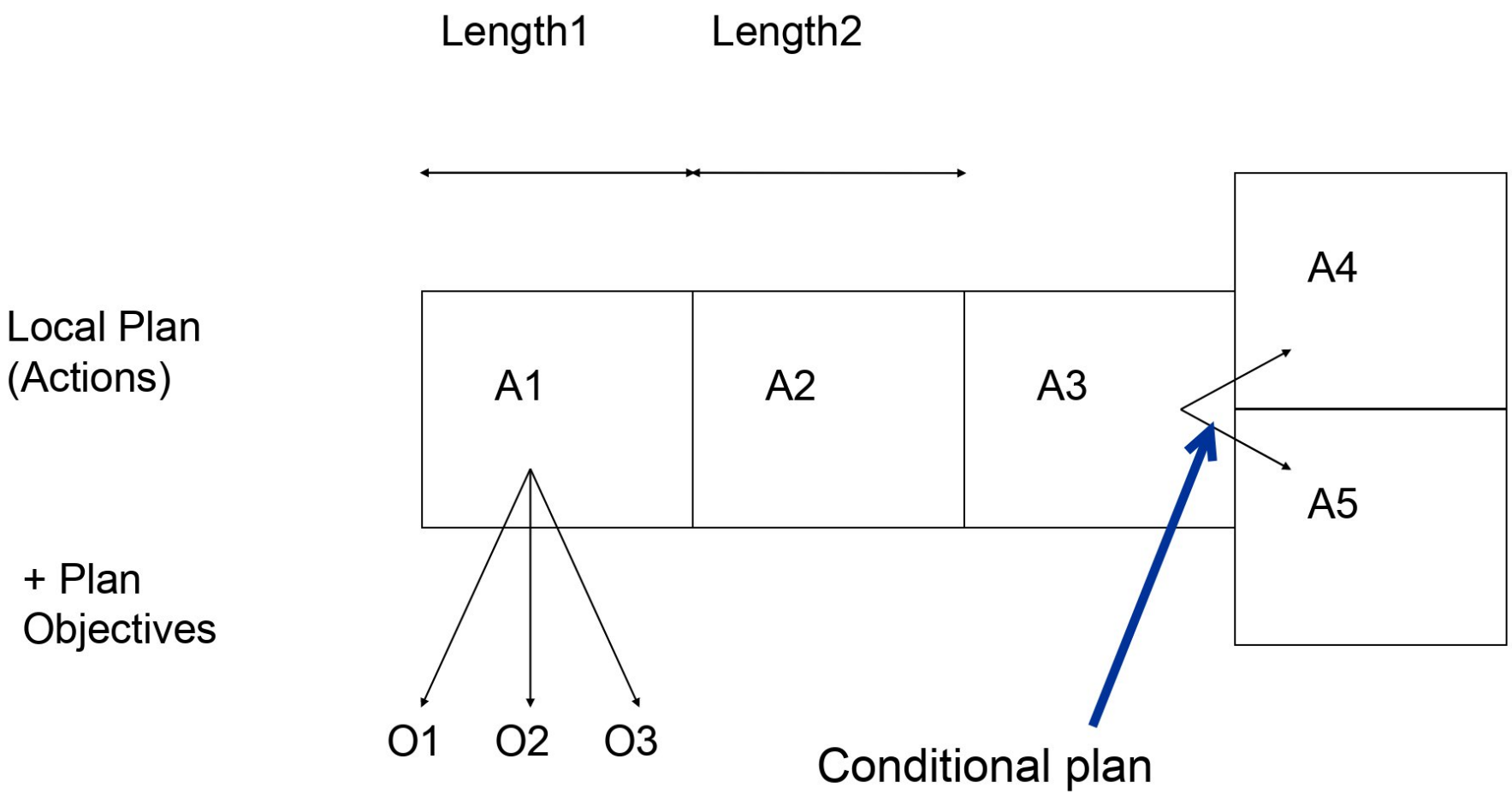
   Each agent may perform an activity to solve its assigned tasks

   The plan can store other information like derived objectives …

   Plan has to be easily modifiable at run time

- The agent knows the duration of its actions
- The agent can determine the quality of its actions

PGP: Create local plans – Schematic view



PGP: Create local plans – Ordering actions

- Prefer actions that concurrently achieve multiple goals
- Prefer actions expected to require less resources (especially time)
- Prefer actions that will strongly verify or refute that some goals are worth pursuing
  - E.g., analysis that confirms that a signal follows a vehicle trace detected by a neighbour agent

PGP: Exchange local plans

2. Exchange plans

   The agents exchange information about their local plans with other agents

- Each agent must have knowledge about the *MAS organisational structure*, so that it can decide which information to send to which agents
- It would be very expensive and inefficient to send all the local plans to all the agents in the system!!

PGP: Generate Partial Global Plans

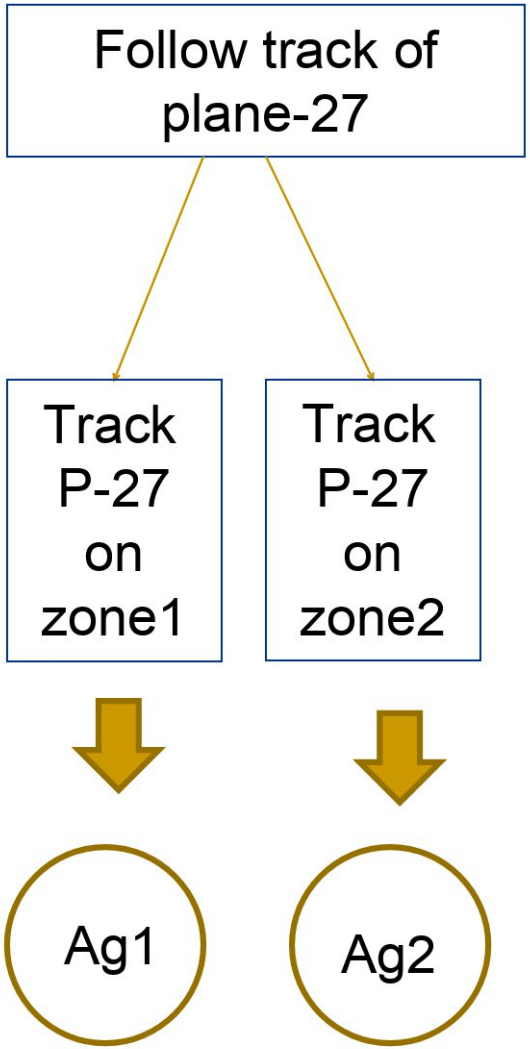
3. Generate *Partial Global Plans* (PGPs)

Each agent models the collective activity of the system by combining the received *local partial plans* into a Partial Global Plan

- Check dependencies between the received information and its own local plan
- Identify when the goals of one or more agents can be considered sub-goals of a single global goal: partial global goal
- Identify opportunities to improve coordination
  - E.g., if two agents need to solve the same sub-problem

PGP: Generate Partial Global Plans - Components

- Partial global goal: final aim of the global plan
- Plan Activity Map: plan actions to be executed concurrently by itself and the other agents, including costs and expected results of actions
  - Initially, it will contain only the union of all the actions of all local plans



PGP: Optimize Partial Global Plans

4. Optimize *Partial Global Plans*

Each agent has a *Planner Module*, specialised in analysing the received information to detect if there are several agents working on the same goal. This information is put in the *Plan Activity Map*, along with the expected future behaviour and expected results of the other agents

PGP: Optimize Partial Global Plans – Building the solution

- From the modified Plan Activity Map, each agent builds a Solution Construction Graph: how the agents should interact, including specifications about
  - What partial results to exchange
  - When to exchange them
  - Who to exchange them with
- It must take into account the estimated time of the actions, their results, etc.

PGP: Optimize Partial Global Plans – Possible Optimizations of Local Plans

- Task reordering
  - Change the order of the actions in the plan
- Task reallocation
  - Move some actions to nodes without assigned work
- Weight of authority
  - Change local plan according to the decisions of the nodes that have more authority
    - This information is obtained by analysing the local plans of nodes with higher authority

PGP: Benefits

- Systems with highly dynamic behaviour
  - All plans can be adapted to dynamic changes in the environment => flexibility
  - However, if an agent changes its local plan, it has to inform other agents (e.g. those that were waiting for a partial result)
- Efficiency
  - If different agents work on the same/similar sub-problems, they will notice that fact in their local plans and reassign their tasks appropriately

PGP: Drawbacks

- Complexity, management of data structures, agents with reasoning capabilities

- Initial version exclusively adapted to the Distributed Vehicle Monitoring problem
  - Generalisation: GPGP

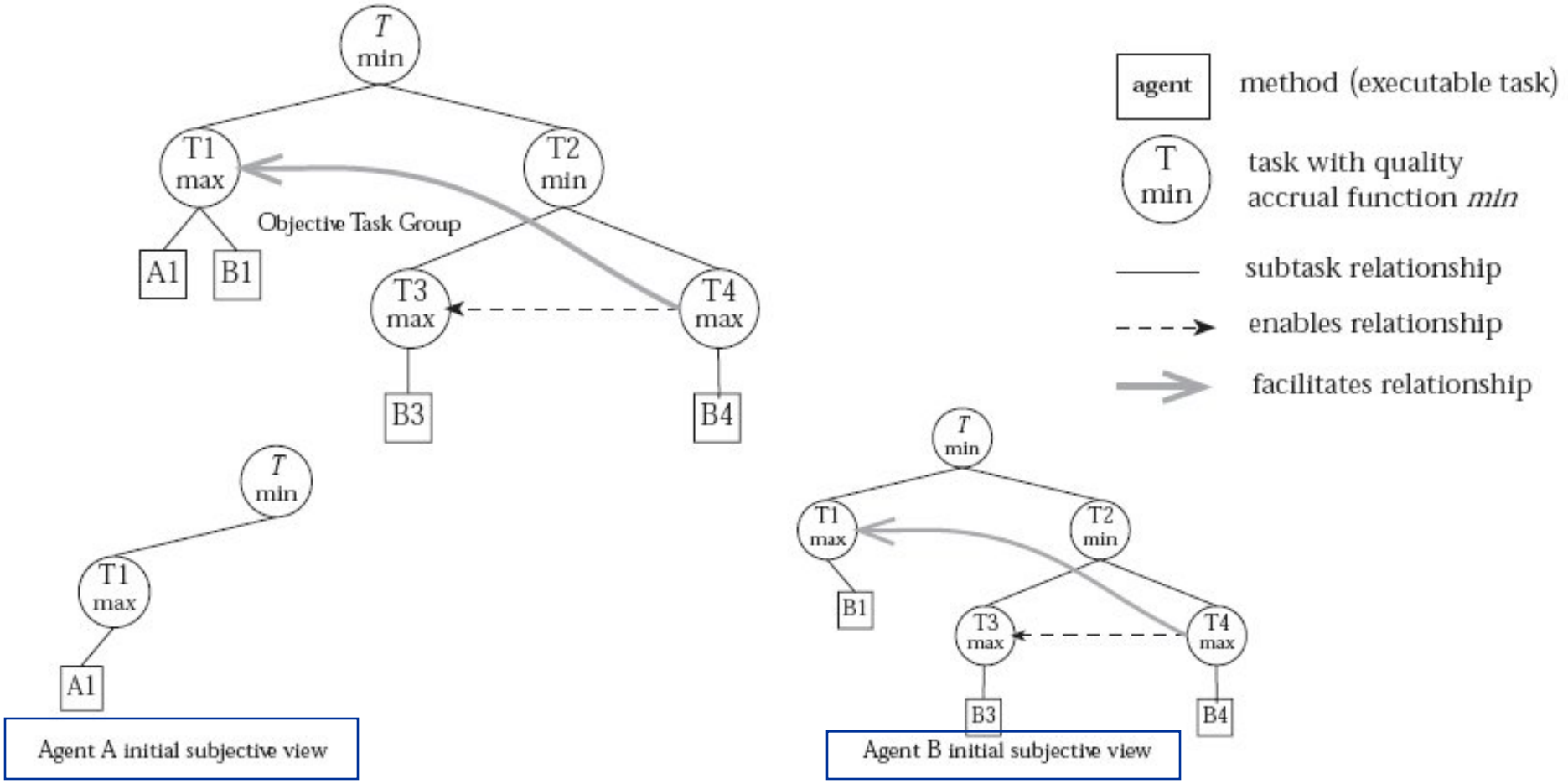
Generalised Partial Global Planning (GPGP) – Decker & Lesser

- Set of modular coordination mechanisms
- Each mechanism responds to certain features of the problem tasks, posing scheduling constraints (to the own agent or other agents)
- Mostly domain-independent (except initial determination of *tasks interrelationships*)

GPGP: Tasks relationships

- A enables B: A must be executed before B
- A facilitates B: executing A makes it possible to solve task B more quickly and with more quality
- A hinders B: executing A implies that solving B will take more time and the solution will have less quality
- The general task structure is an AND-OR tree
  - OR nodes: redundancy
  - AND nodes: sub-problems

Example (I)



Min = AND node Max = OR node

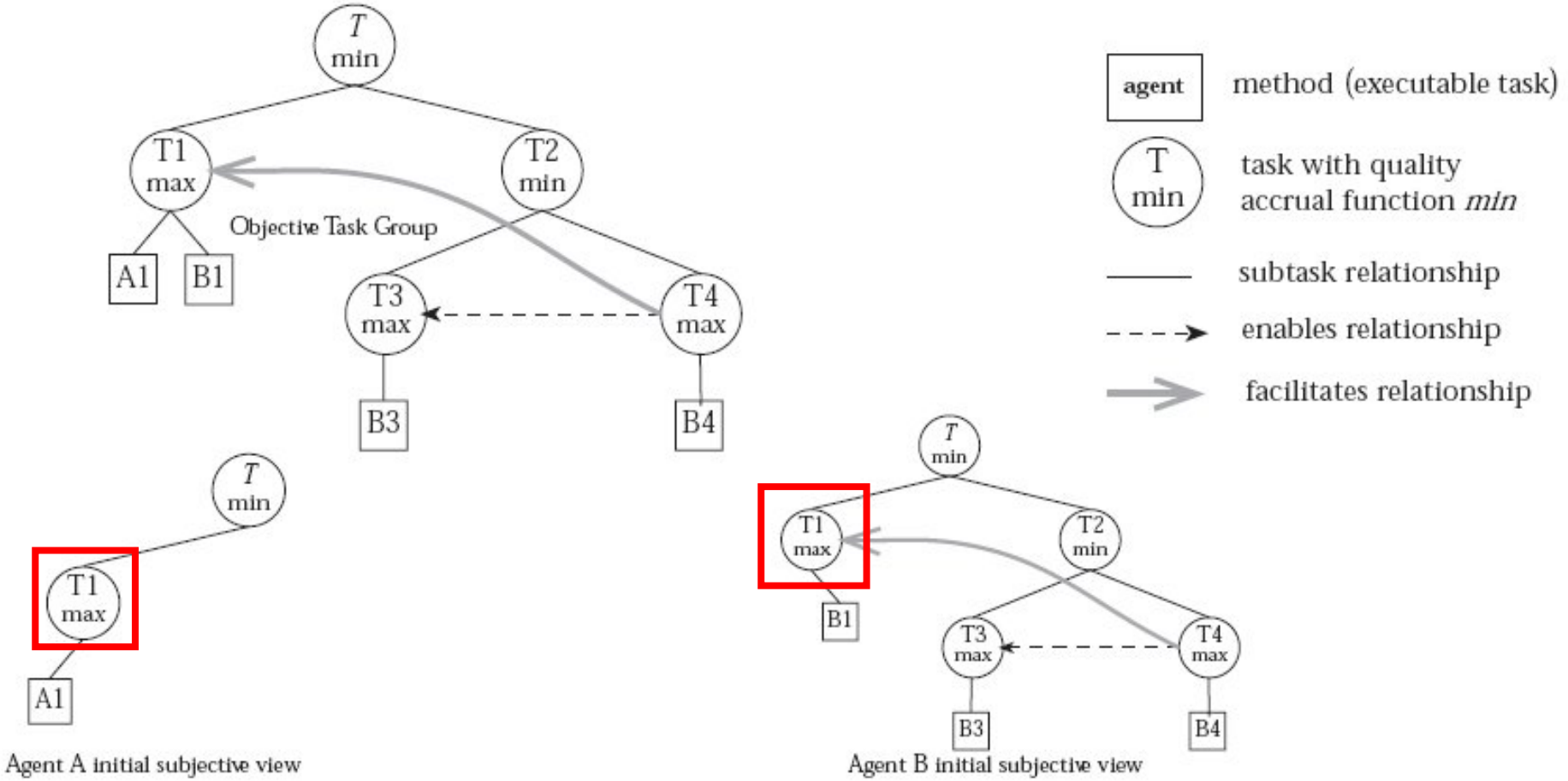
GPGP: Agent structure

- Belief database
  - Information about the task structure
- Coordination mechanisms (CMs)
  - Analyse the task structure and the relationships between tasks to determine scheduling restrictions, and make commitments based on them (a specific task should be finished before a certain deadline with a given quality)
- Local scheduler
  - Constructs a possible schedule based on the local and non-local commitments

CM: Update non-local viewpoints

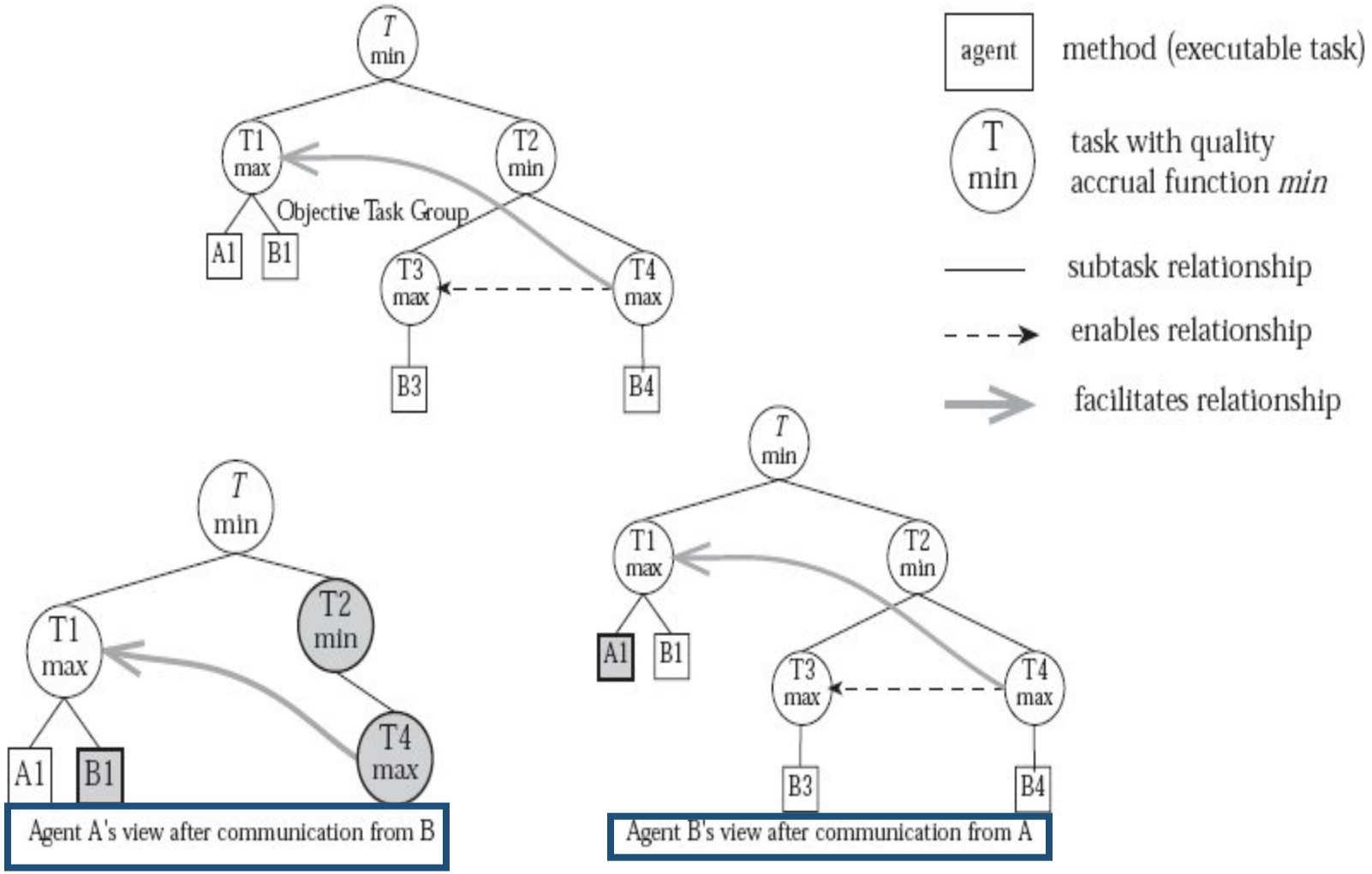
- Each agent sends information about a task T that it can solve to other agents that are also capable of solving it
- Actions that solve the task, relationships with other tasks

Example (IIa)



Task T1 may be solved by both agents

Example (IIb)


T
min
T1
max
T2
min
Objective Task Group
A1
B1
T3
max
T4
max
B3
B4
agent
method (executable task)
T
min
task with quality
accrual function *min*
subtask relationship
enables relationship
facilitates relationship
Agent A's view after communication from B
Agent B's view after communication from A

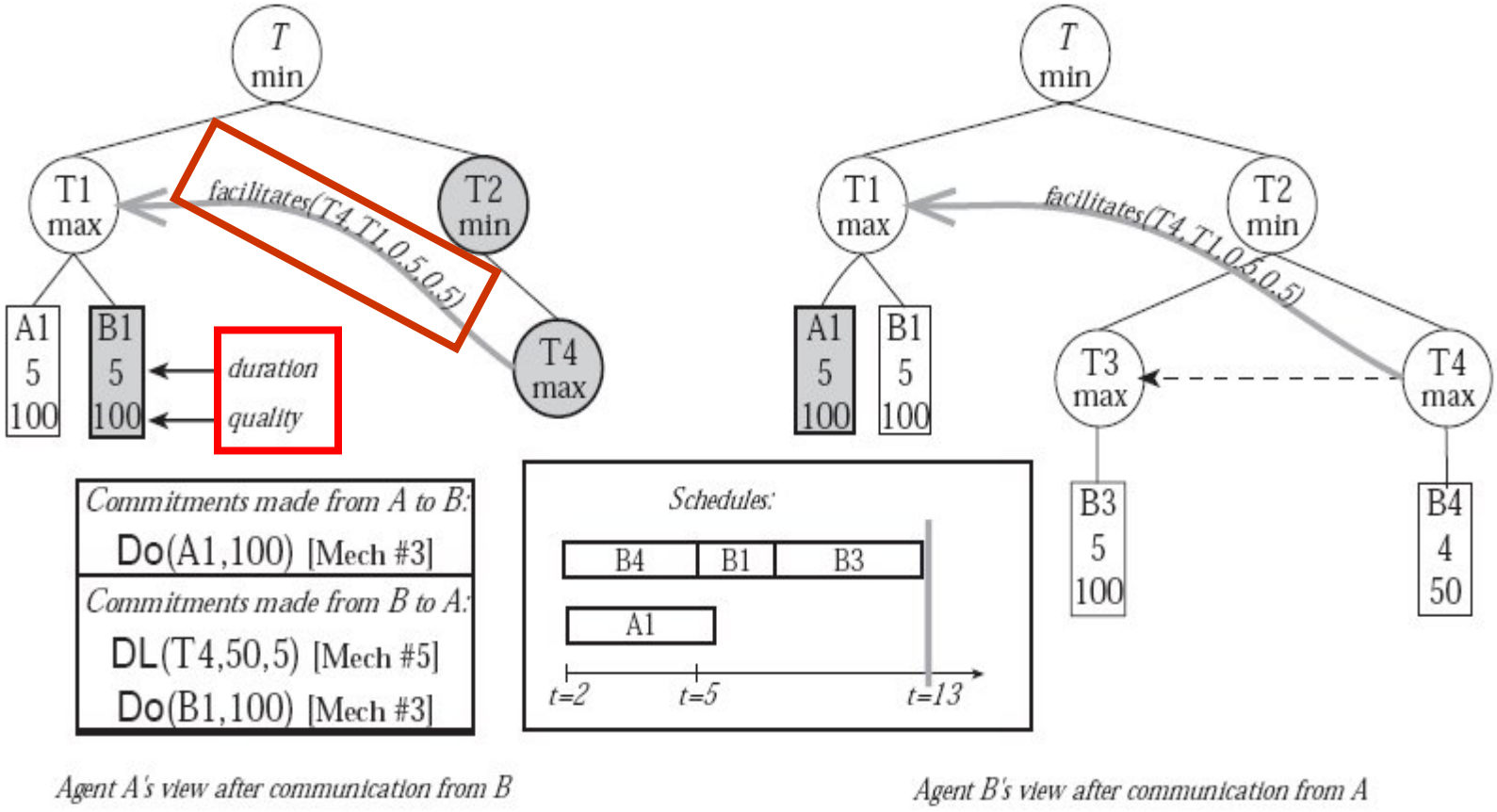
CM: Management of hard coordination relationships

- If A enables B, the agent that can make an action to complete A makes a commitment to finish A as early as possible (so that it gives the maximum space for B to be scheduled – by the same agent or other agents)

CM: Management of soft coordination relationships

- If A facilitates B, an agent can make a commitment to finish A as early as possible, if it is possible (so that it gives to B the possibility to be solved faster and with better results)
- If A hinders B, an agent can make a commitment to finish B as early as possible, if it is possible (so that its execution time and solution quality are not disturbed by solving A)

Example (III)



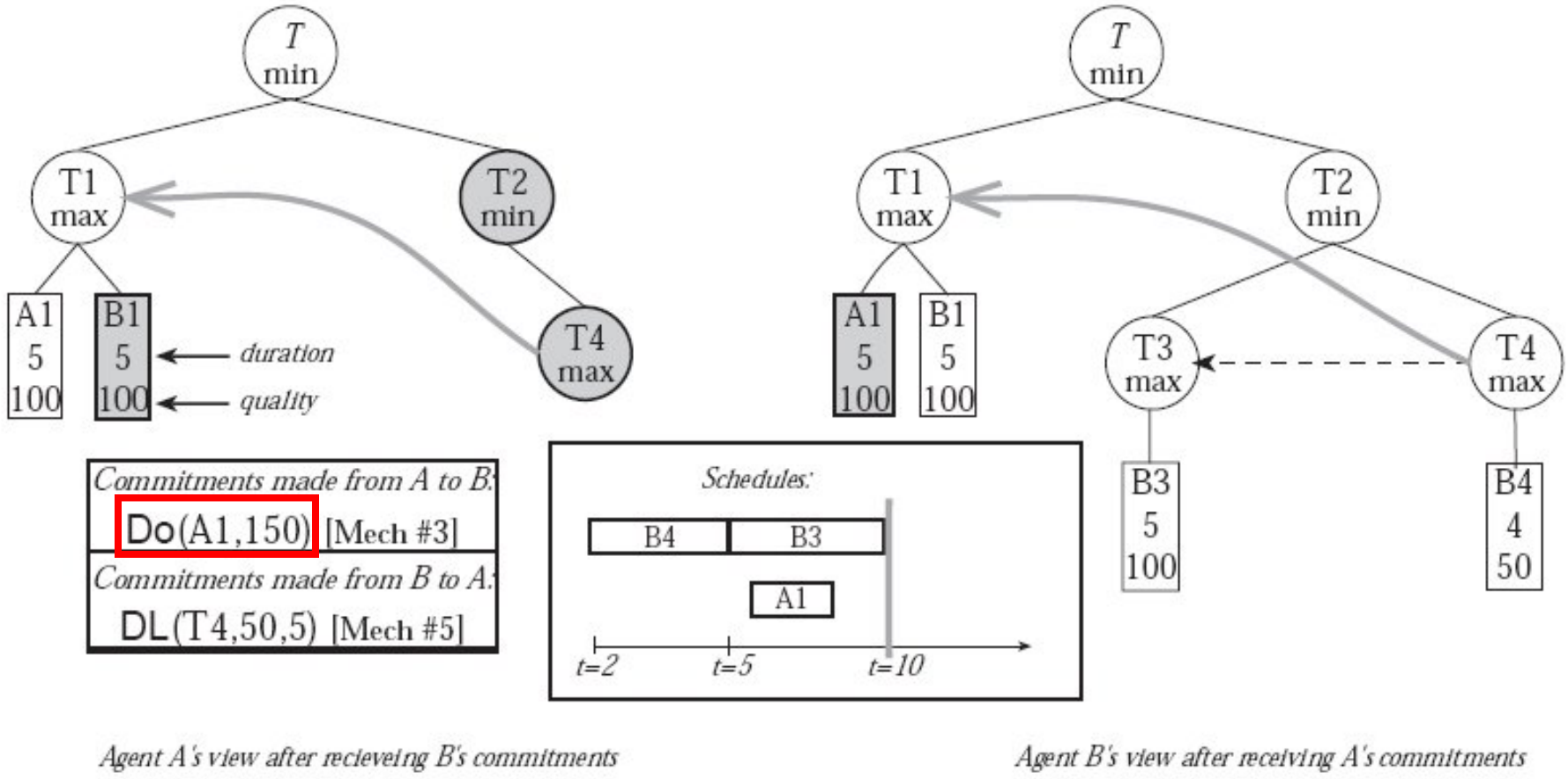
T4 facilitates T1 and enables T3 => solve T4 as soon as possible

Commitment types: Do(Action,Quality), DL(Task, quality, deadline)

CM: Redundancy detection

- If two agents commit to actions that solve the same task, they apply a common algorithm to decide who takes the task and who abandons it
- Options:
  - Random choice
  - Choose agent that obtains solution faster
  - Choose agent that obtains a better solution
  - Choose agent that has less assigned tasks

Example (IV)



Agent A's view after recieveing B's commitments

Agent B's view after receiving A's commitments

GPGP: Conclusions

- Using the commitments information and the schedules, the agents agree in the distribution of the tasks and the final plan
- The different coordination mechanisms can be adapted to each problem by using different policies (different modules)
- Managing dynamic problems can be easy with this method

Proposed readings

- Chapter 8 of the book *An introduction to MultiAgent Systems* (M. Wooldridge), 2nd ed.
- Chapter 4 of the book *Agentes Software y Sistemas Multi-Agente* (A. Mas)
- Paper on cooperation hierarchy (Doran et al.)
- Paper on Distributed Partial Global Planning (Decker & Lesser)