多智能体协作（Work Together）

毛文吉

中国科学院自动化研究所

On Agents Work Together

- Important to make a distinction between:
  - *Benevolent* agents

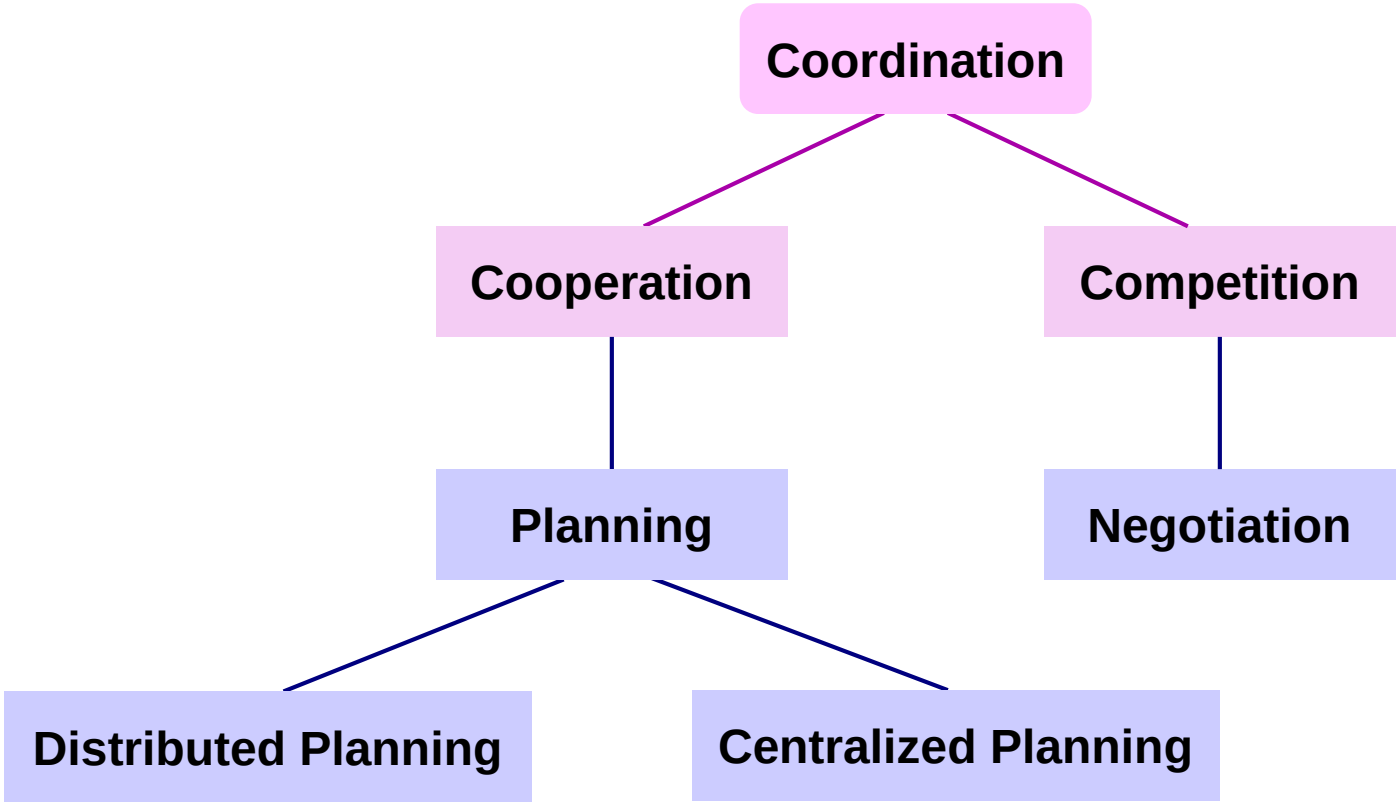
    Problem-solving in benevolent systems is *cooperative distributed problem solving* (CDPS)
  - *Self-interested* agents

    Agents are assumed to act to further their own interests, possibly at expense of others; Potential for *conflict*

➢ Benevolence *simplifies* system design task enormously, *while* self-interest may *complicate* the design task

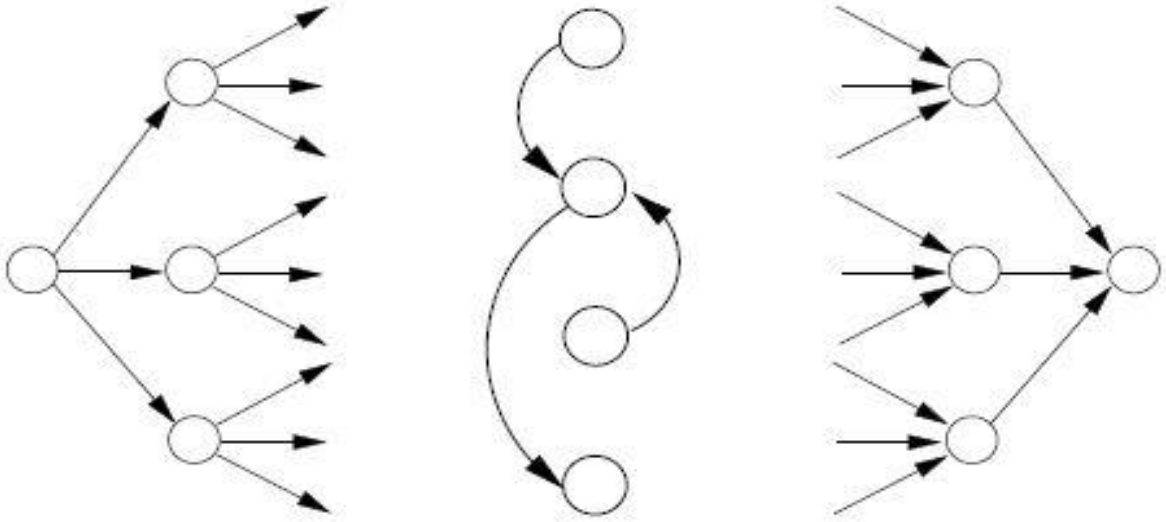
多智能体协作的方式

- A taxonomy of some of the *different ways* in which agents can coordinate their behavior and activities



Cooperative Distributed Problem Solving

- Solving problems through the cooperation of multiple agents. *Three* stages of CDPS (Smith & Davis, 1980):



Problem decomposition | **Sub-problem solution** | **Solution synthesis**

- Control and data are distributed
- No agent has sufficient information to solve entire problem

Problem Decomposition

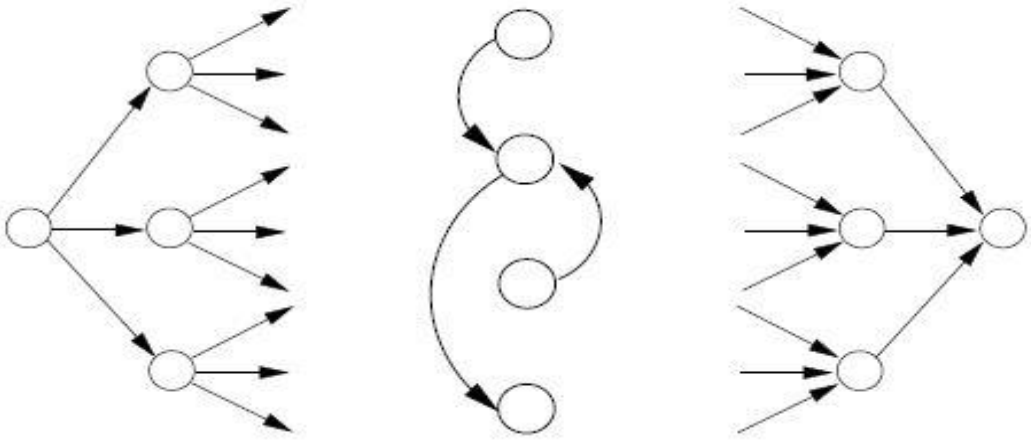
- The overall problem to be solved is divided into smaller *sub-problems*
- Typically a *recursive/hierarchical* process

Design choices:

- *How* is this decomposition process done?
- Is it *centralized* or *distributed*?
- *Who* has the knowledge of task structure?
- Which agents are going to solve the *sub-problems*?

Sub-problem Solution

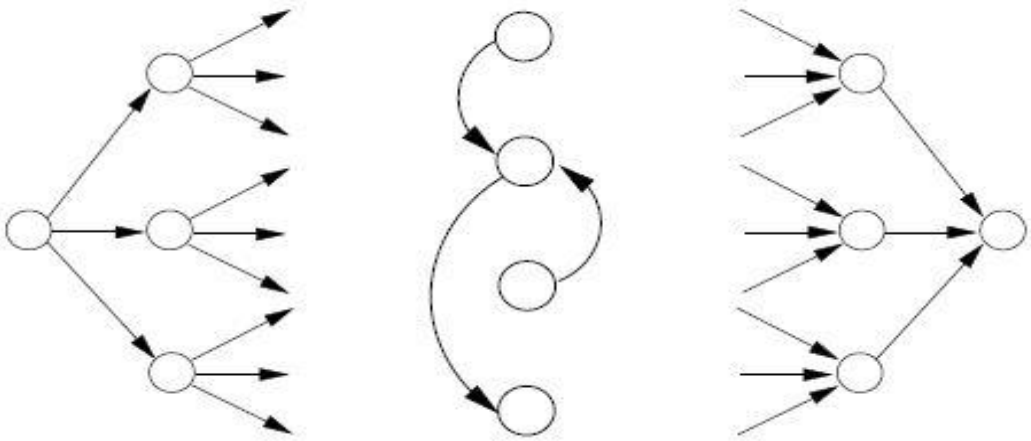
- The *sub-problems* derived in previous stage are solved
- Typically agents *share* some *information* during this process
- May involve two agents *synchronizing* their actions



Sub-problem Solution

Solution Synthesis

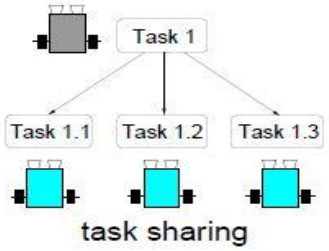
- Solutions to sub-problems are *integrated* in this stage
- Again this may be *hierarchical*
  - Different solutions at different levels of *abstraction*

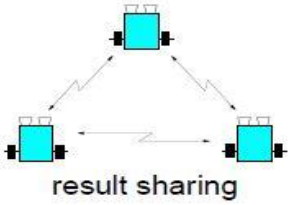


Solution Synthesis

How to Work Together

- Overall, need to be able to *share*
  - Tasks
  - Information





- Two main modes of *cooperative problem solving*:
  - *Task sharing*

    Components of a task are distributed to different agents
  - *Result sharing*

    Information (partial results, etc) is distributed

Fundamental questions:

- How to *allocate tasks* to agents?
- How to assemble a *complete solution* from partial ones?

Outline

- Task sharing using the contact net
- Result sharing in blackboard systems
- Distributed planning via partial global plans
- Distributed vehicle monitoring (DVM)

Outline

- Task sharing using the contact net
- Result sharing in blackboard systems
- Distributed planning via partial global plans
- Distributed vehicle monitoring (DVM)

The Contract Net（CNET, 合同网协议）

- Most well known task-sharing protocol for *task allocation*
- The contract net (CNET) includes *five* stages (Smith, 1980):

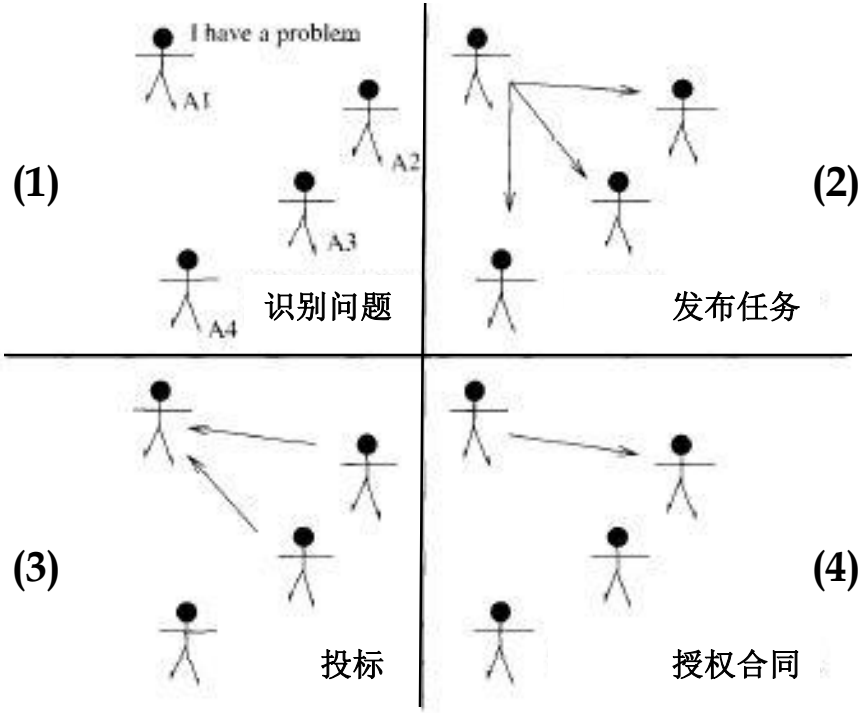
(1) Recognition（识别）

(2) Announcement（发布）

(3) Bidding（投标）

(4) Awarding（授权）

(5) Expediting（完成）



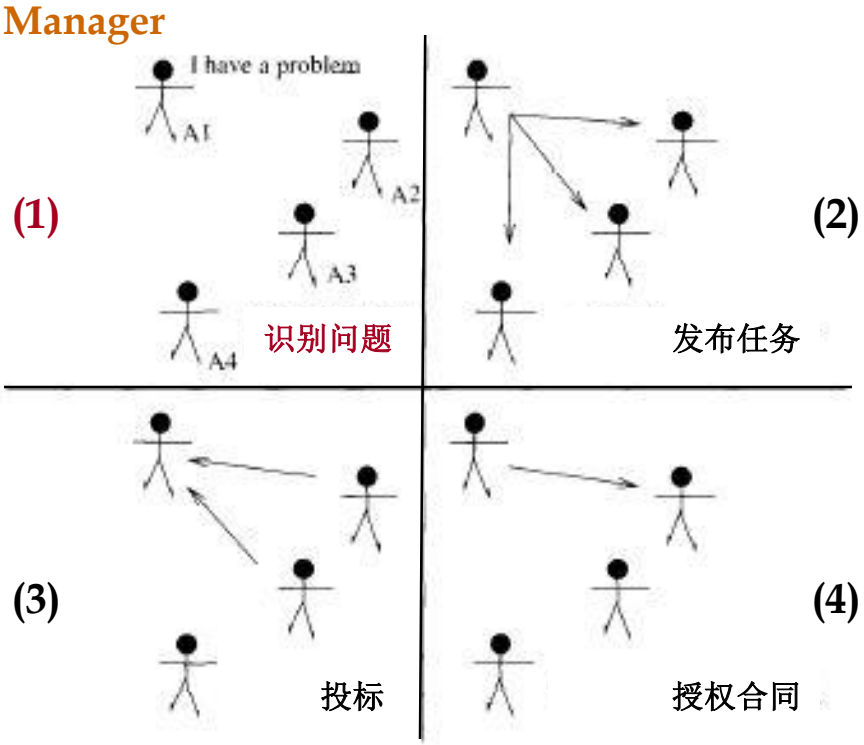
The Contract Net: Stage 1

- The contract net (CNET) includes *five* stages (Smith, 1980):

(1) Recognition（识别）

- An agent recognizes it has a *problem* it wants to help with

  Agent has a *goal*, and either
  - realizes it cannot achieve the goal in isolation – does not have capability
  - realizes it would prefer not to achieve the goal in isolation (typically because of solution quality, deadline, etc)



As a result, it needs to involve *other* agents

The Contract Net: Stage 2

- The contract net (CNET) includes *five* stages (Smith, 1980):

(1) Recognition（识别）

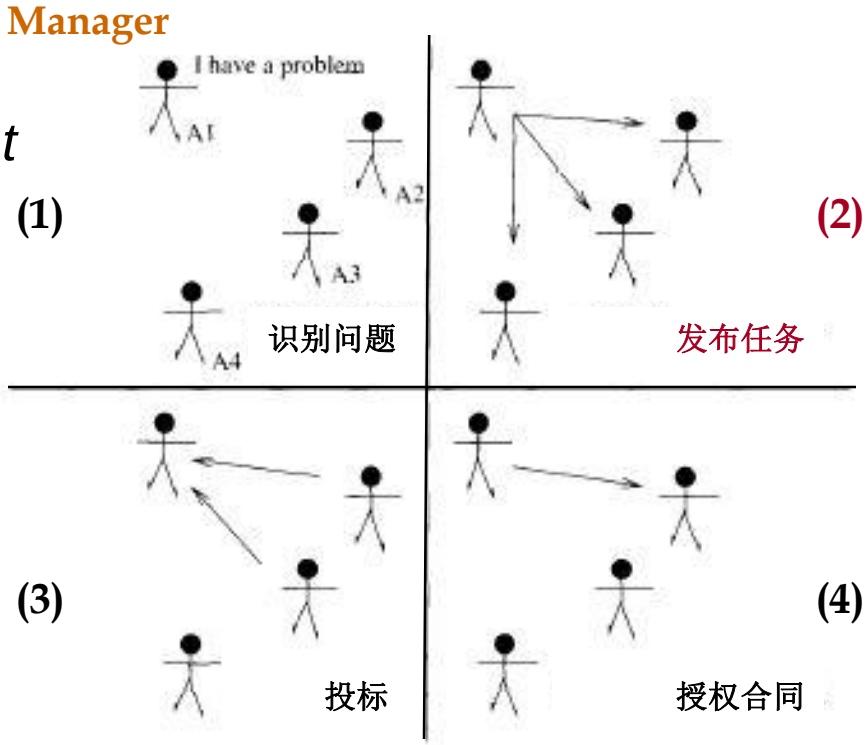
(2) Announcement（发布）

- Agent sends out an *announcement* of the task with its *specification*

  Specification must encode:

  - description of task itself
  - any constraints (e.g. deadlines, quality constraints)
  - meta-task information (e.g. “bids must be submitted by…”)

  The *announcement* is then *broadcast*



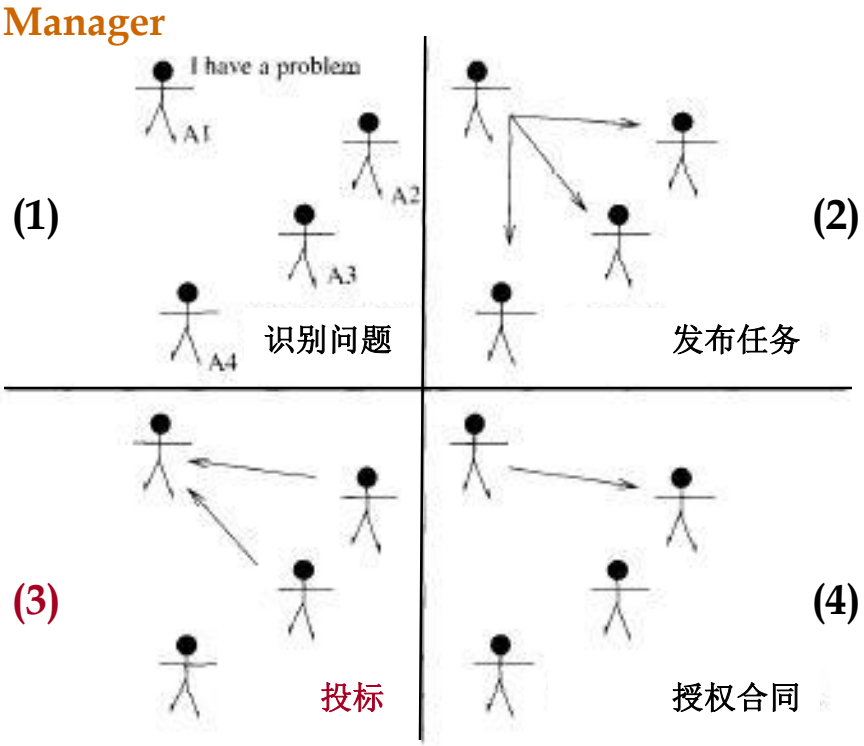
The Contract Net: Stage 3

- The contract net (CNET) includes *five* stages (Smith, 1980):

(1) Recognition（识别）

(2) Announcement（发布）

(3) Bidding（投标）



- Agents decide for themselves whether to *bid* for the task

  Factors:

  - decide whether agent itself is capable of expediting task
  - determine quality constraints & price information (if relevant)

If they choose to *bid*, then submit a *tender*

The Contract Net: Stages 4&5

- The contract net (CNET) includes *five* stages (Smith, 1980):

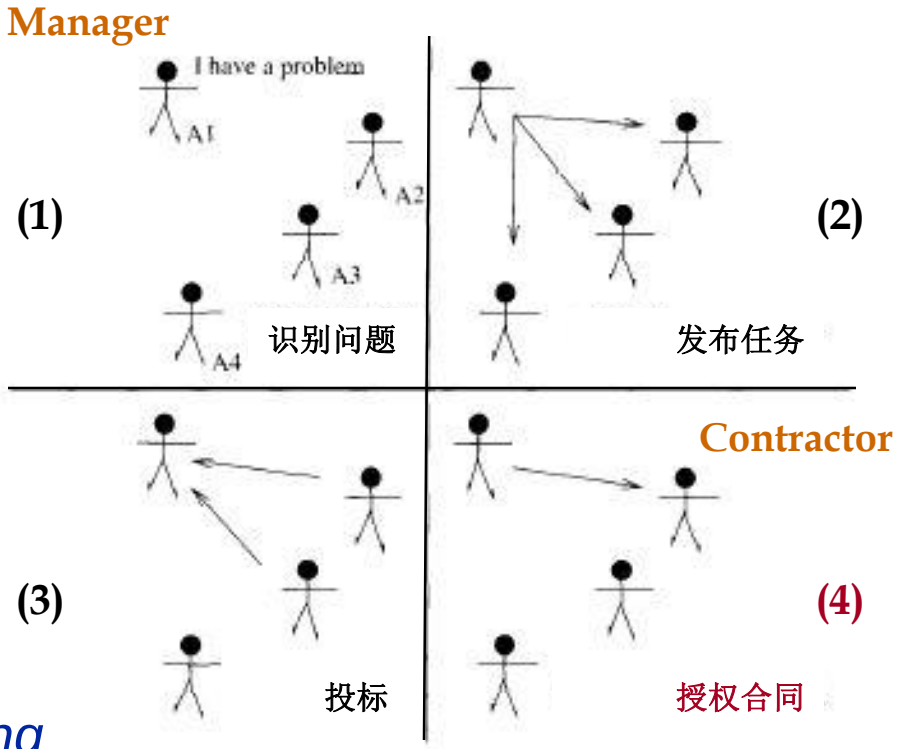
(1) Recognition（识别）

(2) Announcement（发布）

(3) Bidding（投标）

(4) Awarding（授权）& Expediting

- choose between bids & decide who to "*award* the contract" to
- result is *communicated* to agents that submitted a bid
- successful *contractor* then *expedites* the task
- may involve further *sub-contracting* (i.e. another contract net)



The Contract Net: Implementation

Message types:

- Task announcement
- Bid
- Award
- Interim report (on progress)
- Final report (including result description)
- Termination message (if manager wants to terminate contract)

The Contract Net: Implementation

- FIPA agent communication language (ACL) is able to capture the contract net

FIPA performatives:

- *call for proposals* (*cfp*):

  Used for *announcing* a task

- *propose*:

  Used for making a *proposal*

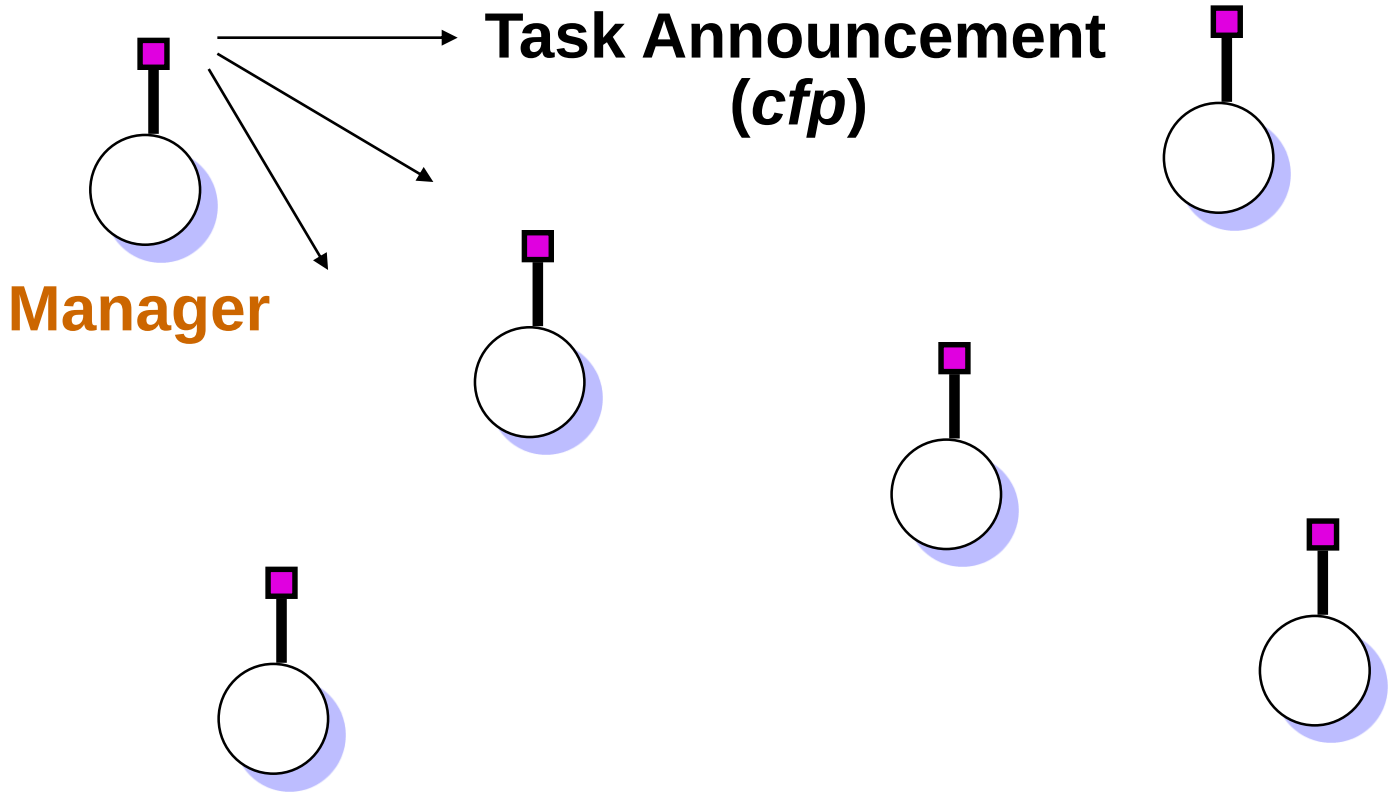
- *accept-proposal*, *reject-proposal* (or *refuse*):

  Used to indicate *acceptance* or *rejection* of a proposal

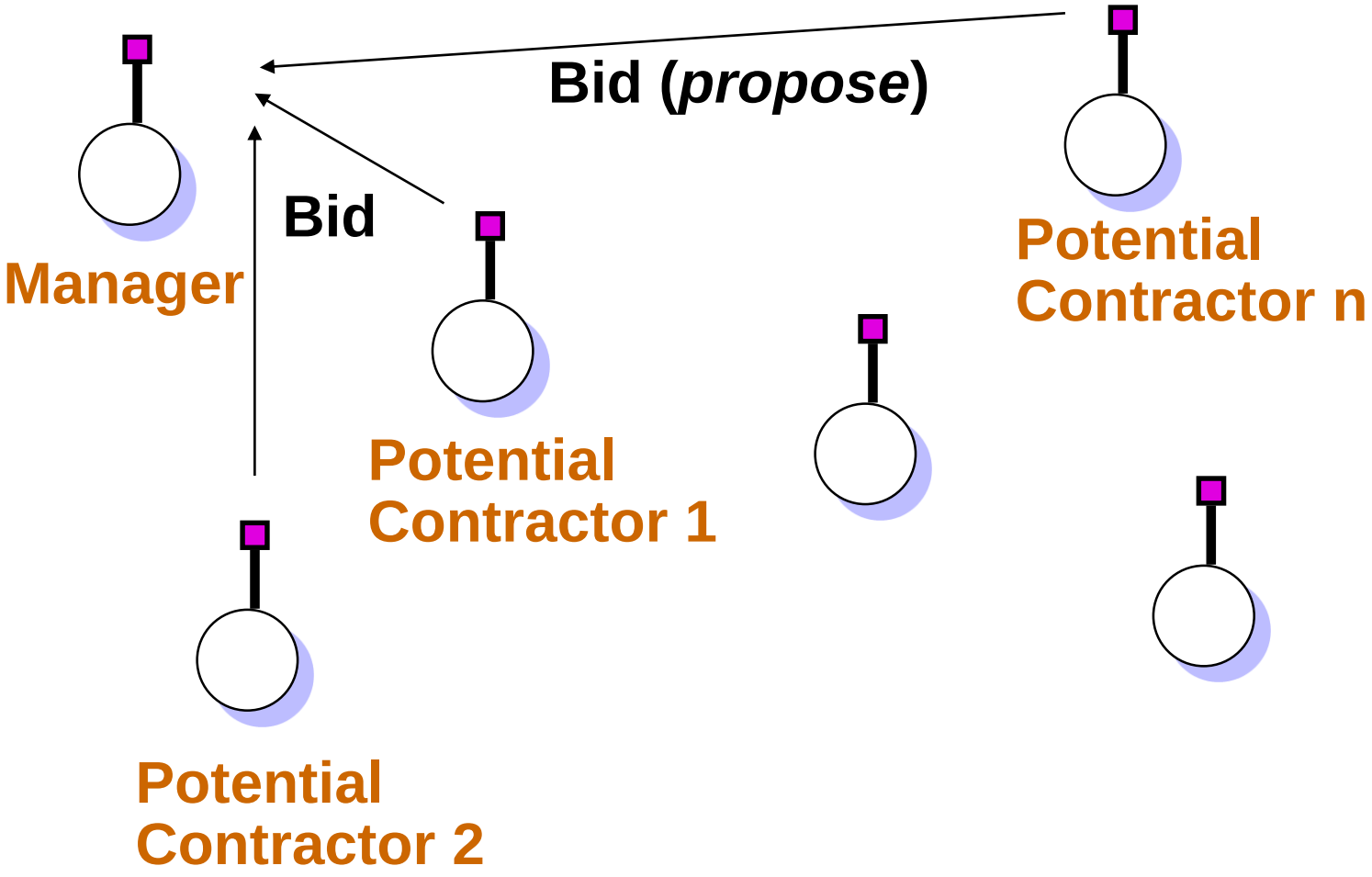
- *inform*, *failure*:

  Used to *inform* completion of a task (with the result) or *failure*

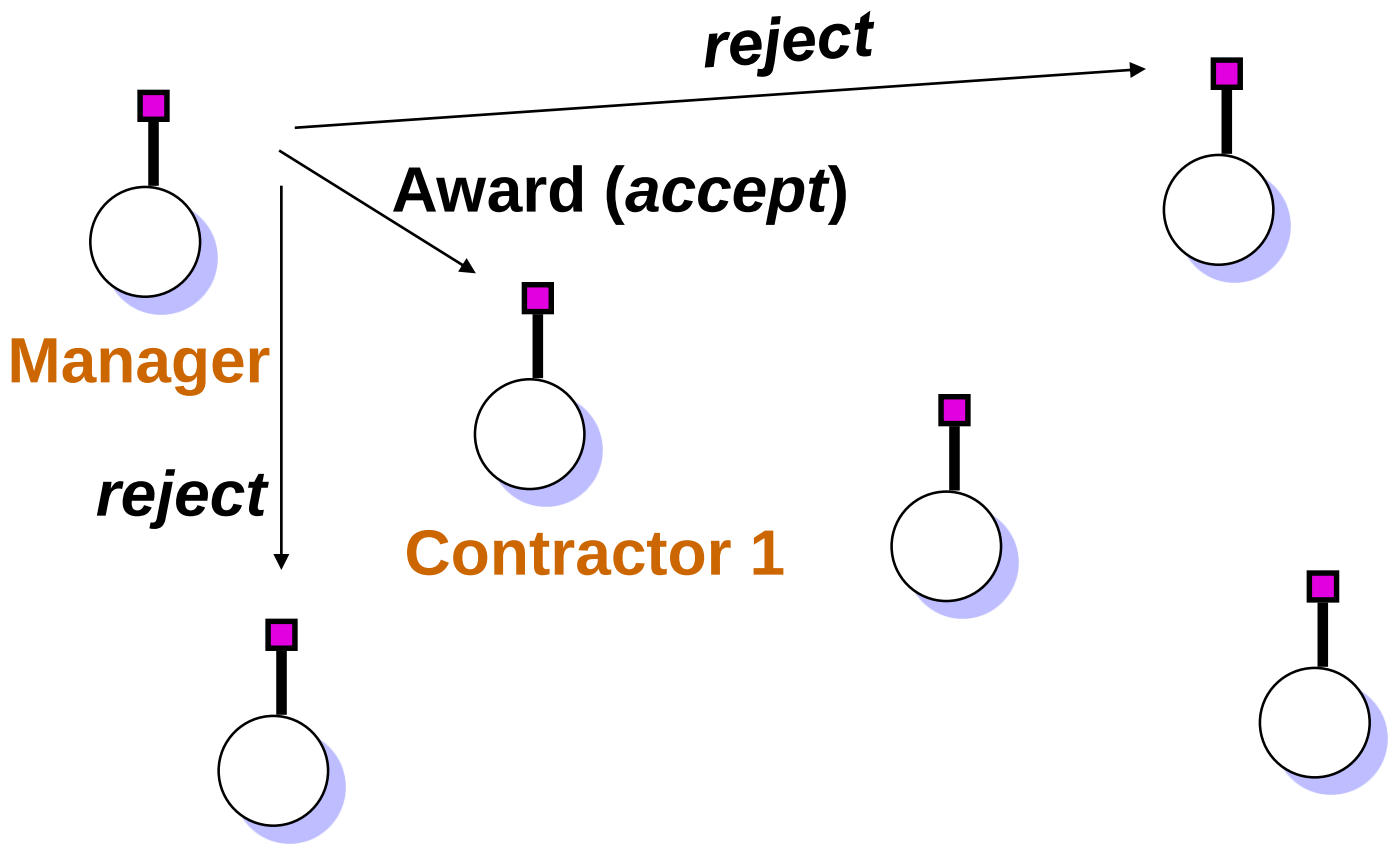
Task Announcement



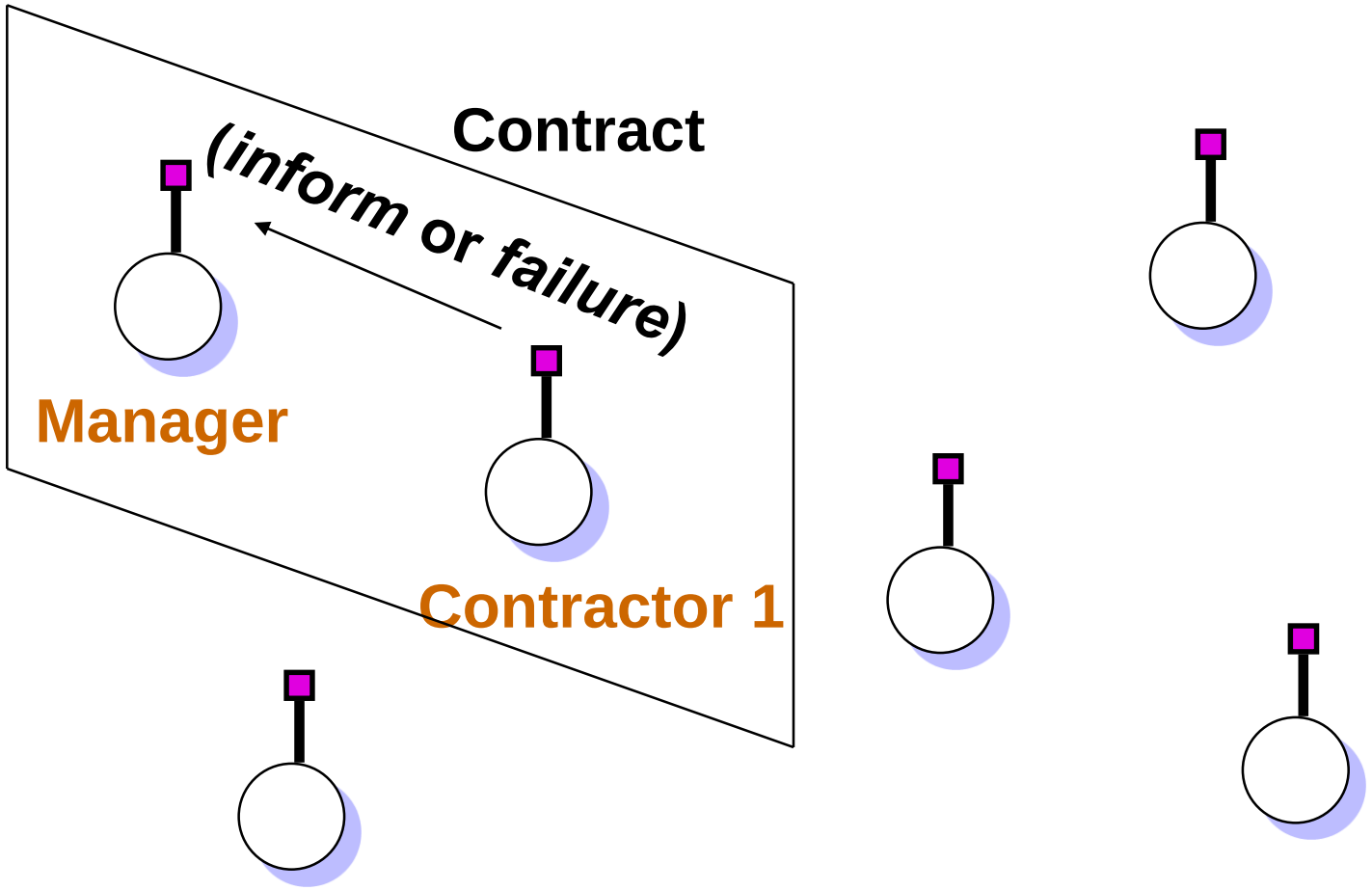
Submitting Bids



Making an Award

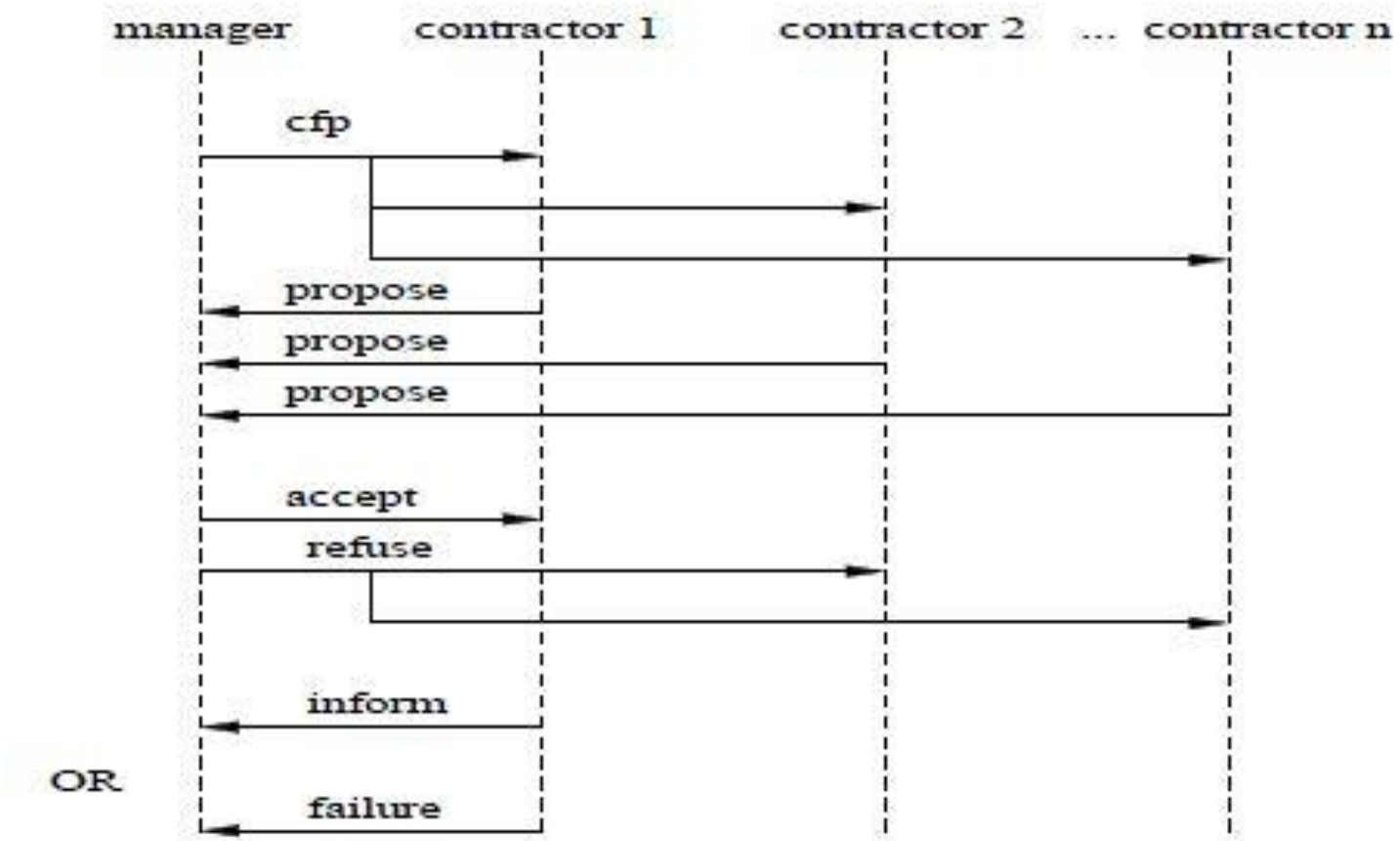


Expediting Task



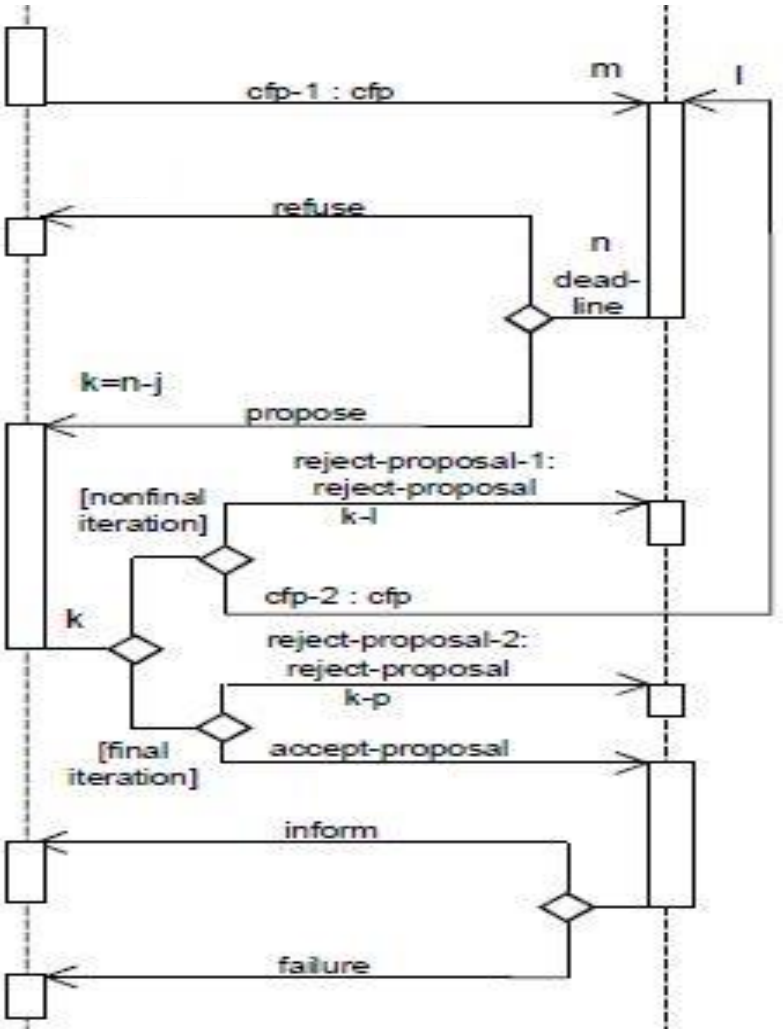
The Contract Net: Implementation

- FIPA agent communication language (ACL) is able to realize the contract net



Iterated Contract Net

- FIPA iterated contract net interaction protocol

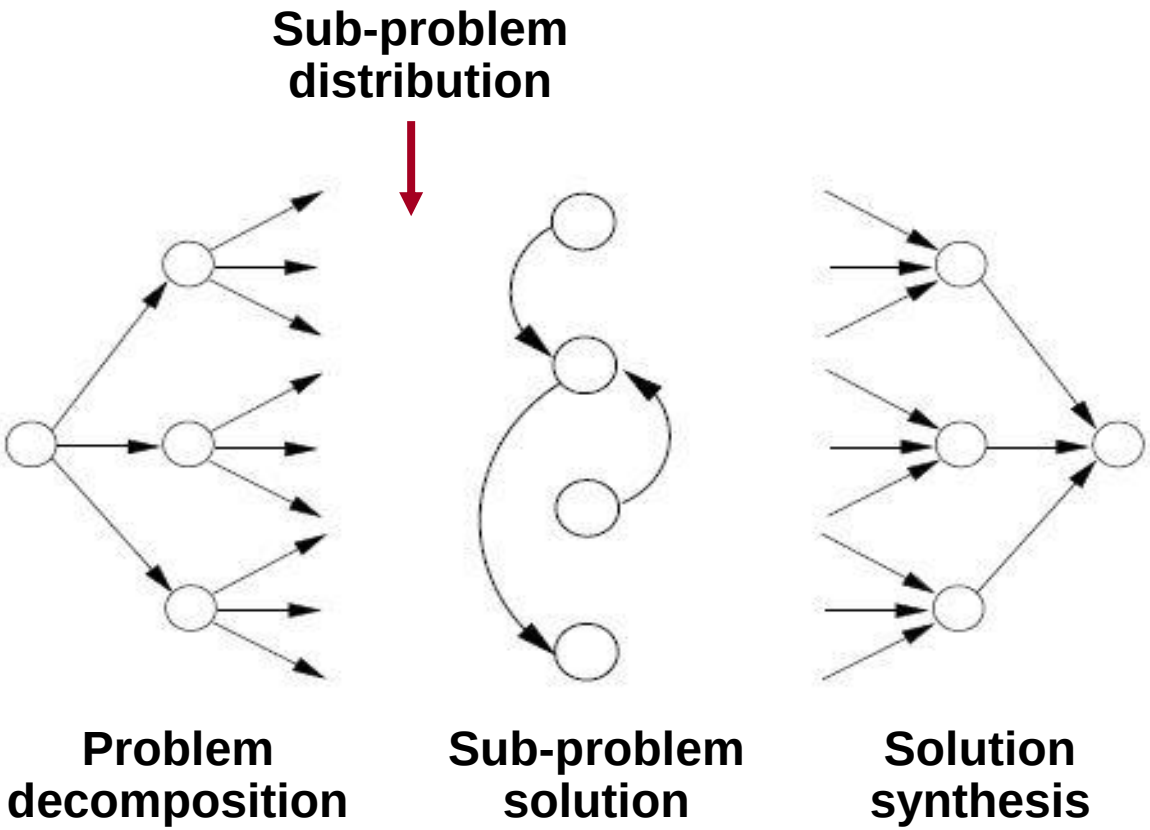


The Contract Net: Features

- Focusing on task distribution for *distributed problem solving*
- Task distribution can be viewed as a kind of *contract negotiation*
- Protocol specifies *content* of communication, not just its form
- Two-way exchange of information and mutual selection are a natural *transfer of control* mechanism
- Despite (or perhaps because of) its simplicity, CNET has been the *most implemented* and *best studied* framework for distributed problem solving

The Contract Net: Limitations

- *Four* stages of CDPS in the contract net
- *Other* stages of distributed problem solving are nontrivial



The Contract Net: Limitations

- Overhead

Efficiency modification:

- *Focused addressing*: when general broadcast isn't required
- *Directed contracts*: when manager already knows which agent is the appropriate contractor
- *Request-response mechanism*: for simple transfer of information without overhead of contracting
- *Node-availability message*: reverse initiative in the negotiation process

Protocol variation:

- *Matchmaker* and *broker*

The Contract Net: Suitable Applications

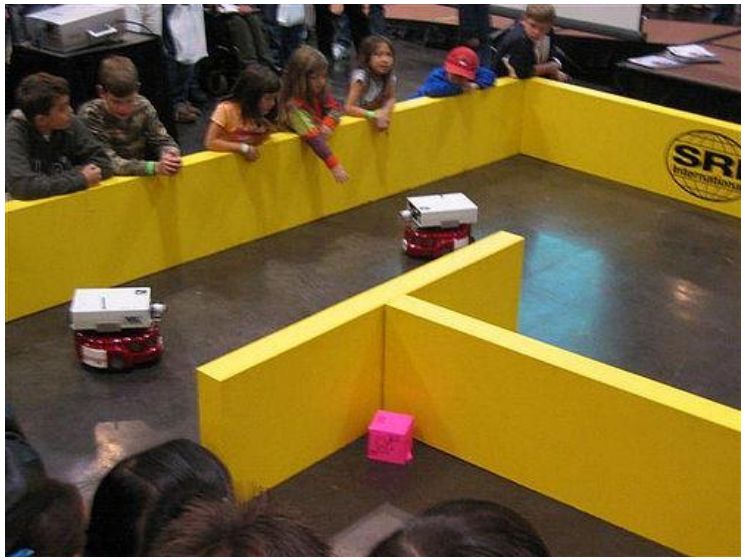
- *Hierarchy* of tasks
- Levels of data abstraction
- Careful selection of *knowledge sources* is important
- Subtasks are *large* (thus it's worthwhile to take effort to distribute them wisely)
- Primary concerns are distributed control, achieving reliability, avoiding bottlenecks

Outline

- Task sharing using the contact net
- ➢ Result-sharing in blackboard systems
- Distributed planning via partial global plans
- Distributed vehicle monitoring (DVM)

Result Sharing

- Agents provide each other with information as they work towards a solution
- Result sharing can improve problem solving by:
  - Independently derived solutions can be *cross-checked*
  - Sharing local views can achieve a *better global* view
  - Sharing results can improve the *precision* of the overall solution
  - By sharing results, a solution can be derived *more quickly*



The Centibots robots collaborate to map a space and find objects

Result Sharing in Blackboard Systems

- The *blackboard system* was the *first* scheme for cooperative problem solving
- Results shared via shared data structure (i.e. the blackboard)
- Multiple agents (KSs/KAs) can *read* and *write* to the blackboard
- Agents write *partial solutions* to the blackboard
- The blackboard may be structured into *hierarchy*

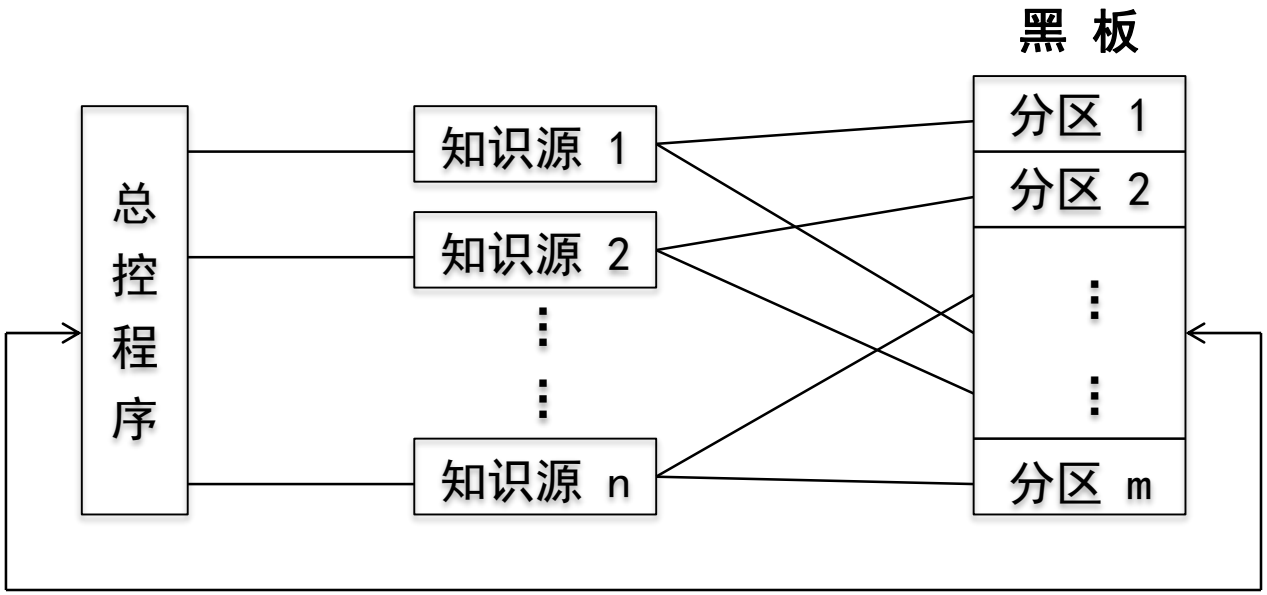
Brief History

- The blackboard architecture was firstly adopted in Hearsay-II speech understanding system (Fennel & Lesser, 1977), a DARPA supported 5 year project
- The complicated structure of speech understanding problem motivated the search for new ways of organizing problem solving knowledge
- *knowledge source* (KS, 知识源)

  An *agent* that embodies knowledge of a particular aspect of problem domain, and takes actions based on its knowledge
  - Speech understanding needs distinct KS's to deal with *acoustic*, *phonetic*, *lexical*, *syntactic* and *semantic* information

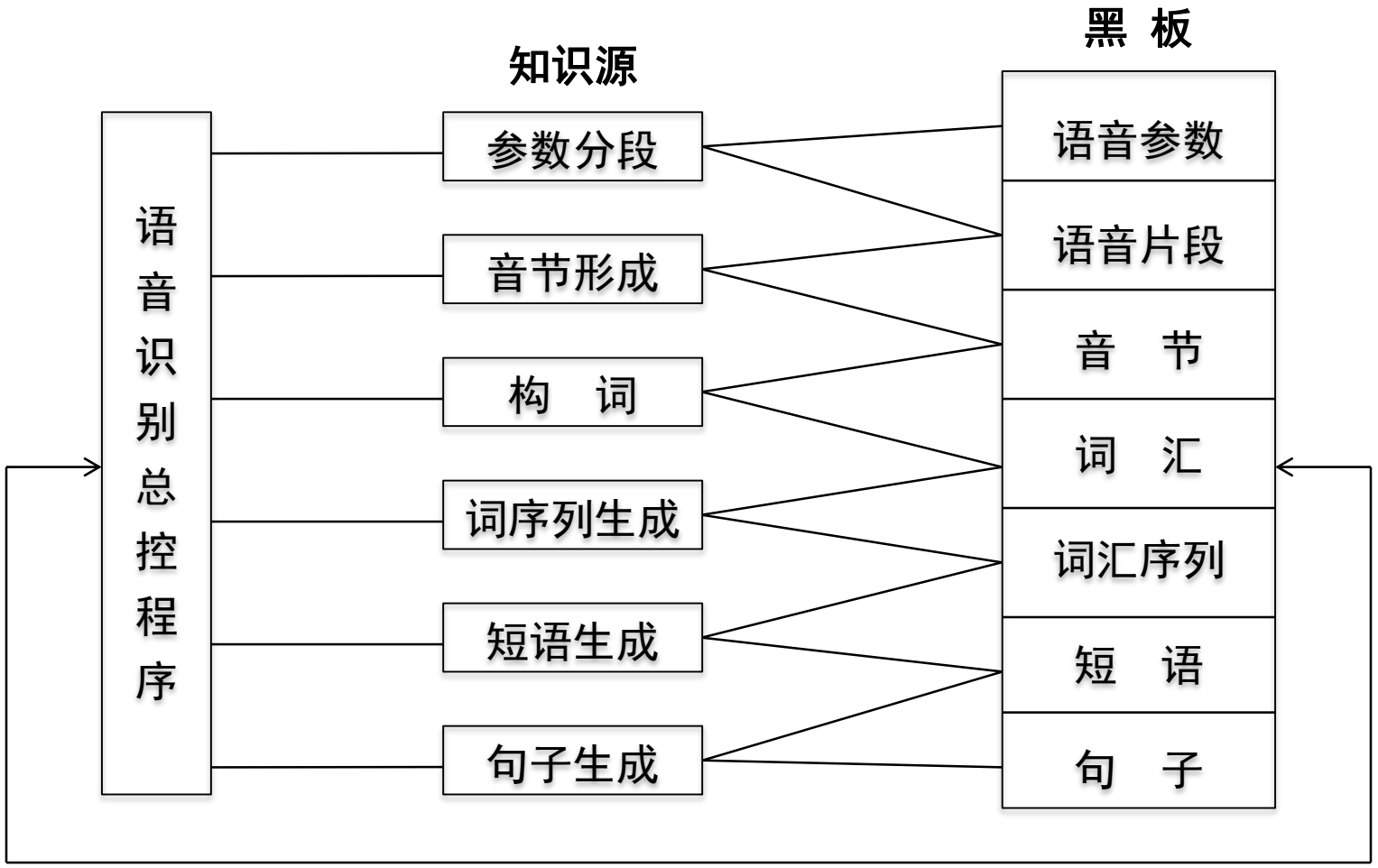
黑板结构

- 将复杂问题求解分解成多个子任务，子任务之间可以异步执行；每个智能体（知识源, KS）独立完成一个特定的子任务，多个知识源通过协作完成问题求解过程
- 黑板是一个信息缓冲区，按信息的抽象程度分层；各知识源通过黑板交换信息，共享数据和结果



HEARSAY-II 口语理解系统（HSII）

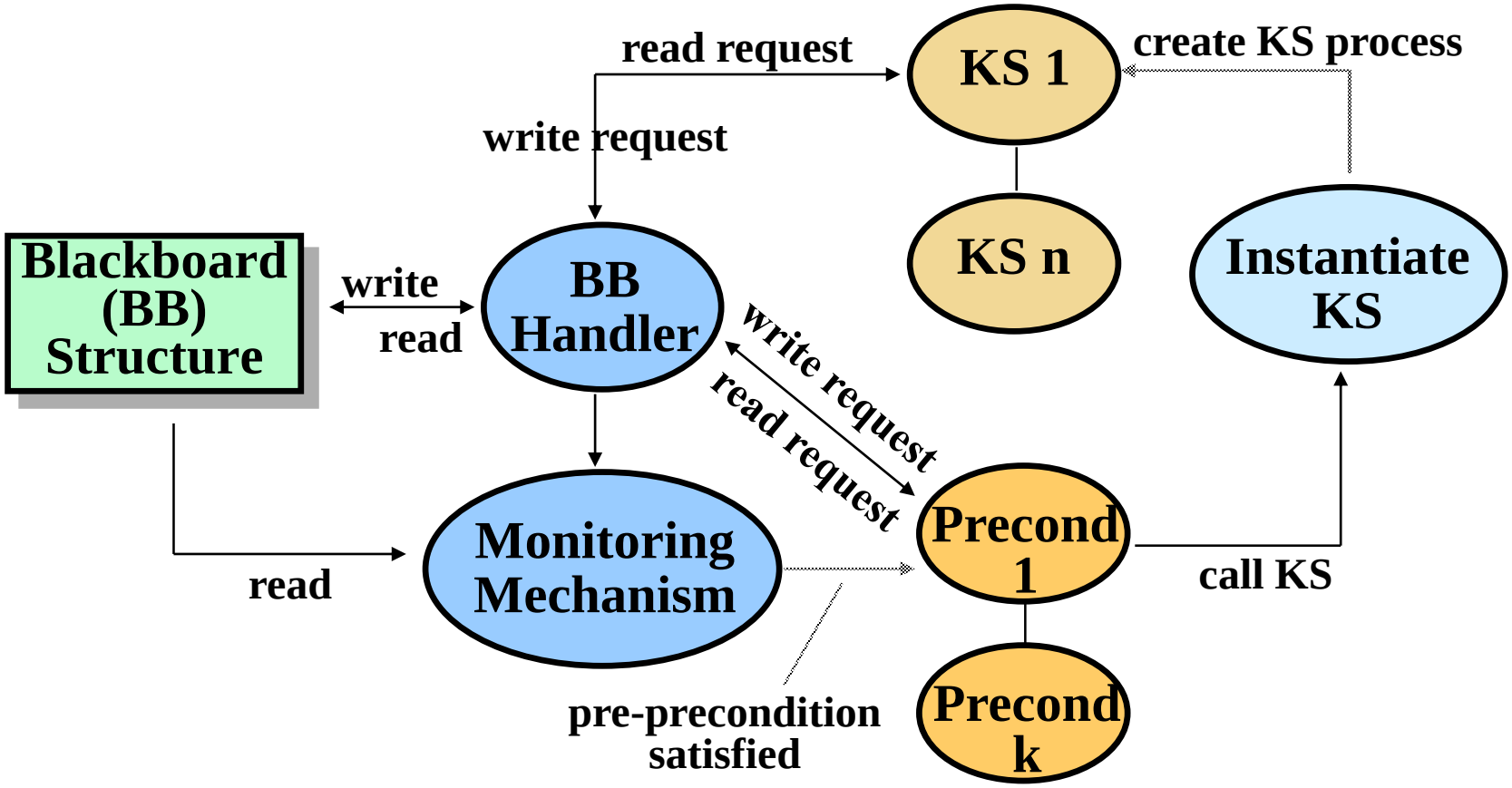
- HEARSAY-II系统的黑板结构：



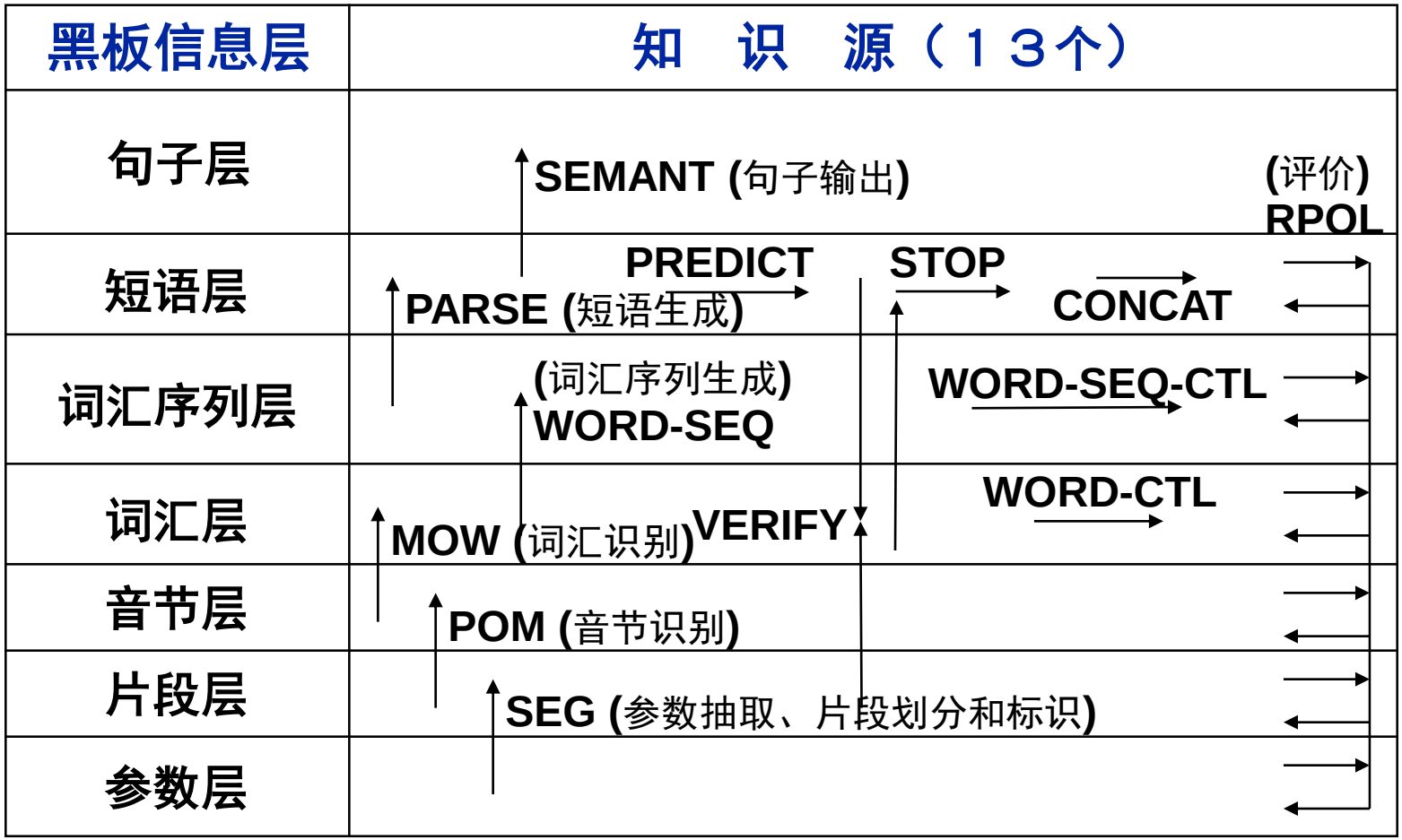
HSII Blackboard Architecture

- The blackboard architecture is a *parallel* production system (Productions: P $\rightarrow$ A)
- *Preconditions* are satisfied by current state of the (dynamic) blackboard data structure, and trigger their associated action (KS process)
- *Actions* presumably alter the blackboard data structure
- Process *terminates* when no satisfied precondition is found, or when a “stop” operation is executed (failure or solution)

HSII System Overview



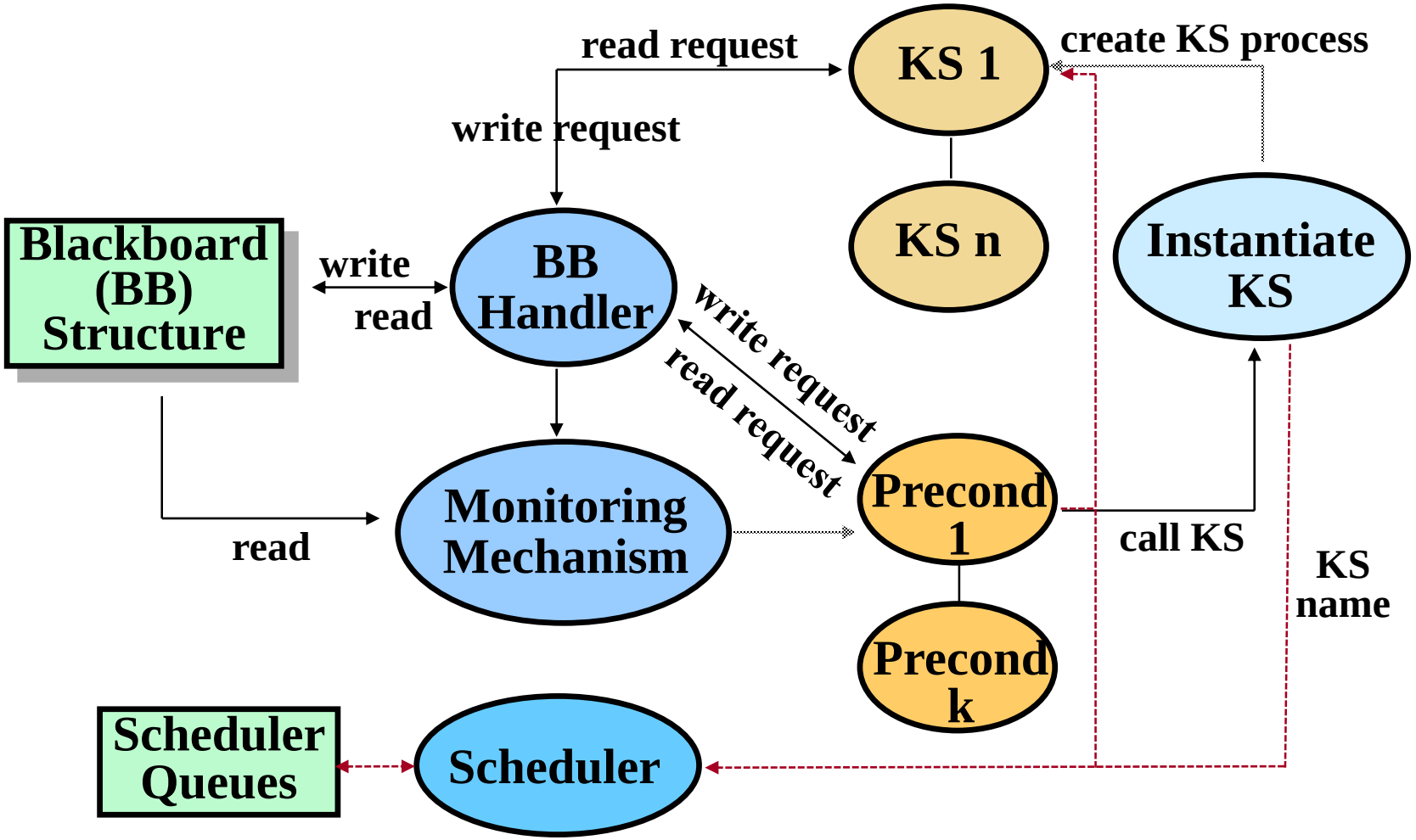
KS and Information Level (Simplified)



HSII Blackboard (Continued)

- Preconditions specify complex tests to decide precondition satisfaction
  - Monitoring relevant data events occurred in BB
  - Precondition tests may perform in parallel
- KS processes generally hypothesize new data, or verify/ modify data already in BB
  - Hypothesize-and-test (生成−测试) problem-solving paradigm
  - *Hypotheses* representing *partial* problem solutions are generated and then tested for *validity*
- Independent preconditions and KS processes potentially generate a *considerable number* of parallel activities
  - Different preconditions, KS's and instantiations of a KS

HSII System Overview



Scheduling Mechanism

- 采用机遇问题求解(*opportunistic problem solving*)策略，通过计算各活动的优先级，确定调度队列的活动执行次序

  例如：

  - 竞争原则：优先执行（优先级）得分最高的活动
  - 正确性原则：优先执行利用最正确数据的KS
    （延缓执行信息不充分的KS）
  - 重要性原则：优先执行其激活框架最重要的KS
    （使KS条件部分成立的一组黑板假设）
  - 功效原则：优先执行最可靠与执行代价最小的KS
  - 目标满足原则：优先执行响应框架最可能满足目标的KS
    （KS动作所产生的黑板状态变化）

➢ 调度机制随机利用最好的数据和最重要最有希望的方法

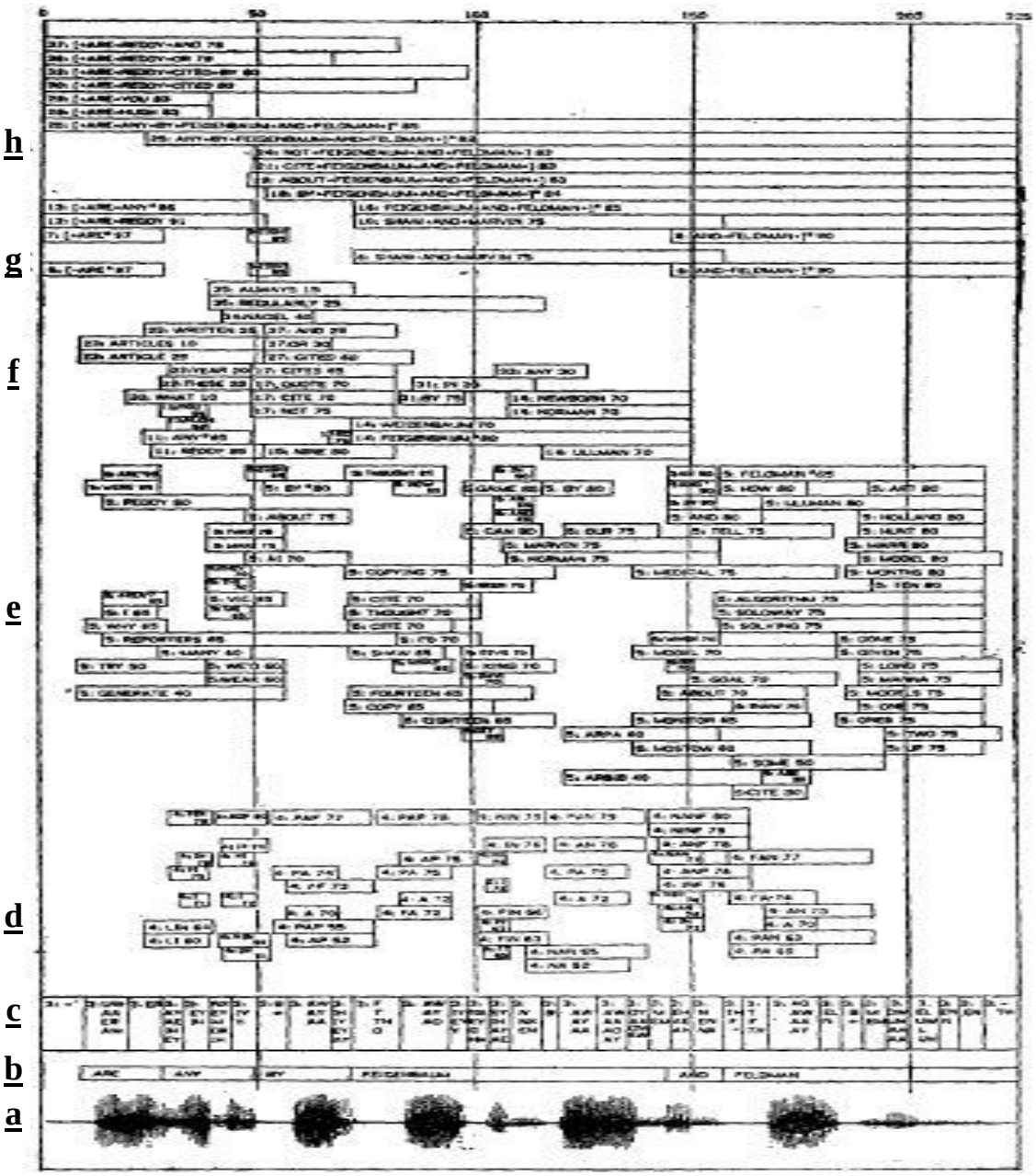
BB Example

- Input sentence

  “Are any by Feigenbaum and Feldman?”

- Utterance time

  2.25 seconds



a–h层：对应输入语音信号、片段、音节、词汇（由MOW生成）、词汇（由VERIFY生成）、词汇序列、短语、句子信息层

KS Modularity and Communication

- Modularity

  The KS's have been developed by many people working in *parallel*

  - It is also useful to check how the system performs using different subsets of KS's

- Communication is indirect

  The KS's are developed *independently* and do not interact with each other directly

  - KS's communicate via the blackboard structure

HSII Experimentation

- Simulation experiments
  - To measure the *software overheads* involved in the design and execution of a complicated, distributed control structure
  - To determine whether there really exists a significant number of *parallel activities* in the speech understanding task
  - To understand how the *inter-process communication* and interference, especially that from data-access synchronization in the blackboard, affect effective parallelism realized
  - To gain insight into the design of an appropriate *scheduling algorithm* for the distributed problem solving structure
- Primary goal *achieved*:
  - 完全正确率：*74%*；语义正确率：*91%*

Blackboard Systems: Main Features

- Unified structure of the blackboard at all information levels
  - Convenient for structuring and *sharing results*
- Integrated representation of alternative hypotheses on BB
  - Avoid recalculation and permit a *global view* of the current state of problem solution
  - Data-directed scheduling policies to support *focus of attention* (注意力聚焦) to the appropriate hypotheses in the BB
- Modularity
  - *Easy to develop* and integrate new KS's into the system, and *reimplement* BB internal structure without KS modification
- Distributed paradigm of problem solving
  - Applied to *diverse domains* in image understanding, reading comprehension, protein-crystallographic analysis, various ES

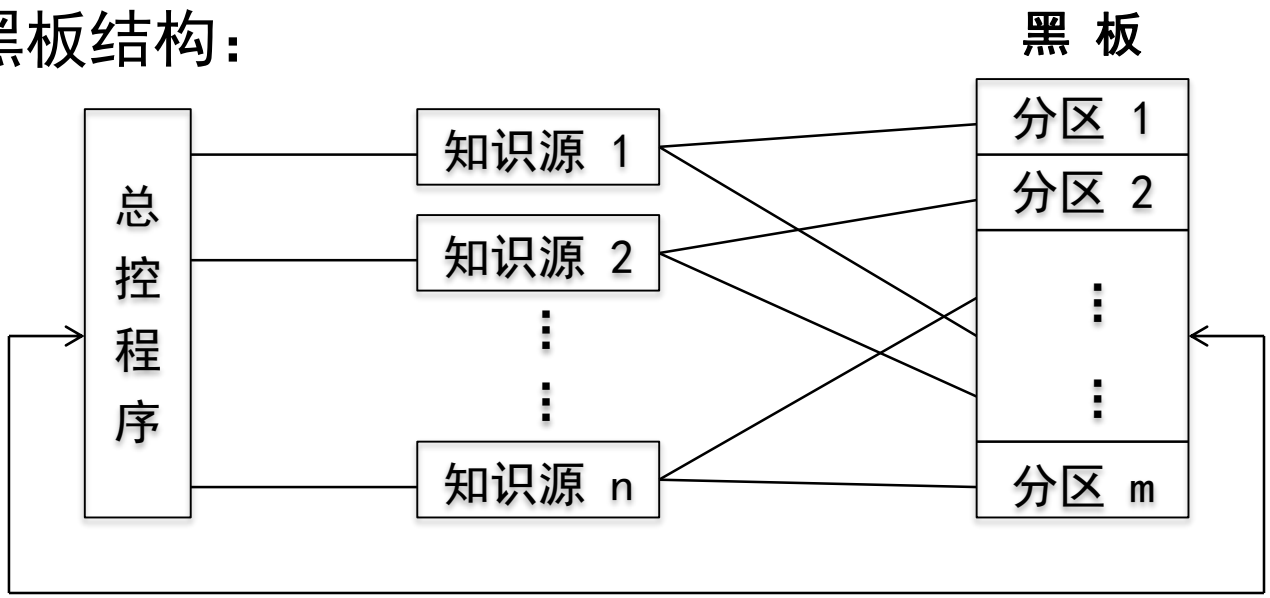
Blackboard Systems: Limitations

- Central control

  Inflexible for sub-problem solver to design local control mechanism

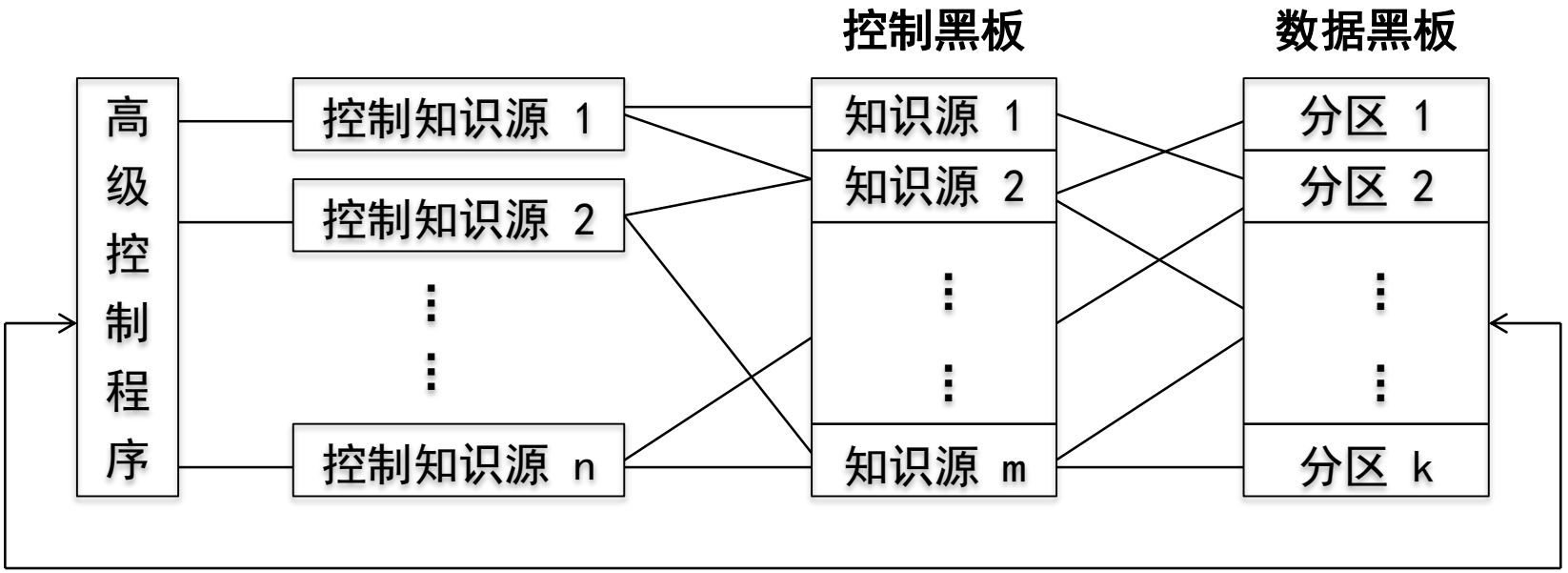
- In fact, Information, processing and control can all be *distributed*

- 单黑板结构：



多黑板结构

- 总控程序由多个分量组成，这些分量称为控制知识源
- 建立一块新的黑板—控制黑板，控制知识源通过这块黑板读取和修改领域知识源
- 高级总控程序驱动控制知识源，控制知识源驱动领域知识源

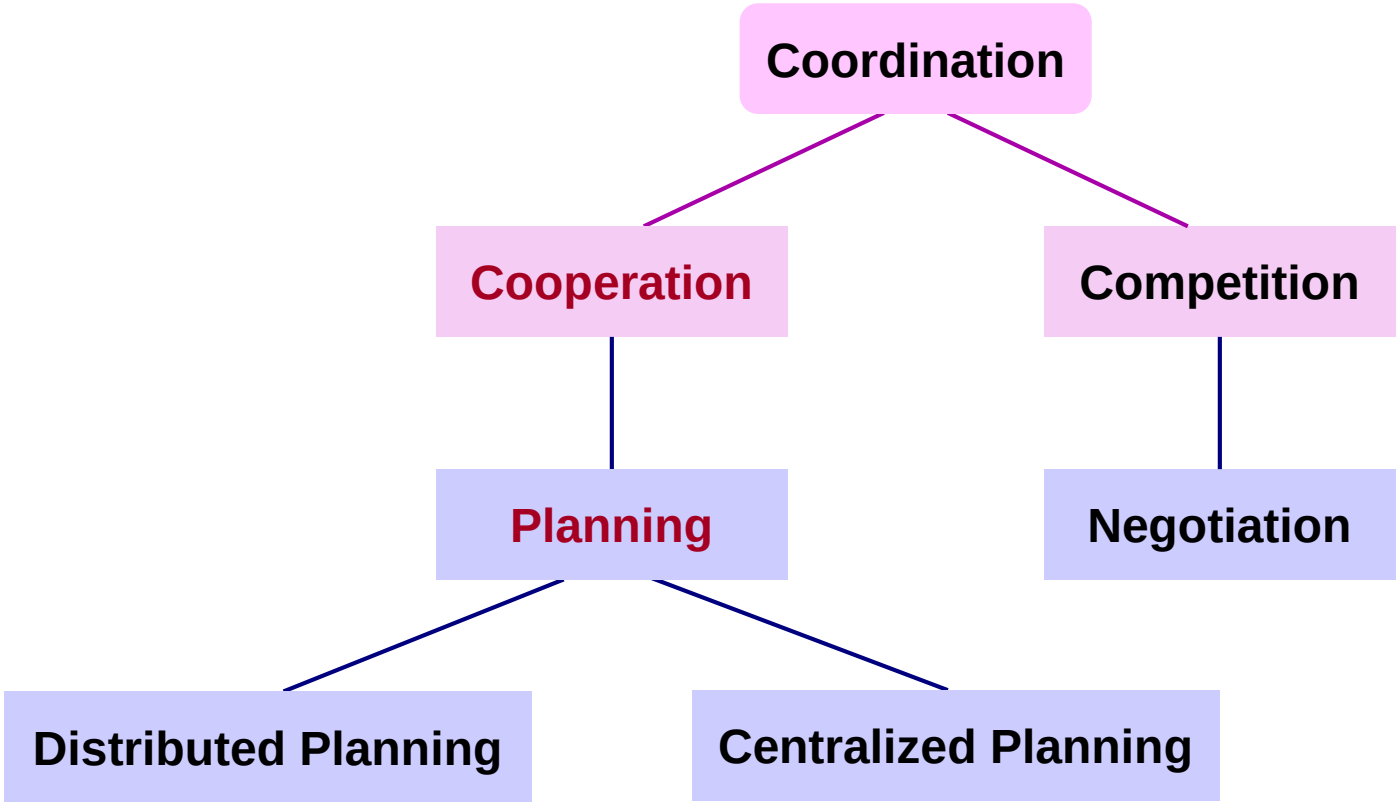


Outline

- Task sharing using the contact net
- Result sharing in blackboard systems
- ➢ Distributed planning via partial global plans
- Distributed vehicle monitoring (DVM)

多智能体规划

- A taxonomy of some of the *different ways* in which agents can coordinate their behavior and activities



Multi-Agent Planning

- Goal is to formulate a plan, but capabilities or expertise to do so is distributed: DPS with heterogeneous agents where the problem is to construct a plan
- Goal is to have a distributed plan, where each agent has its piece of the plan that, in concert with others, achieves goal
  - Can be formed in a *centralized* manner
  - Can be formed in a *distributed* manner

Centralized Planning for Distributed Plans

For exploiting available (homogeneous) agents:

1. Generate a partial-order plan with minimal ordering constraints
2. Decompose the plan into subplans such that ordering constraints between steps tend to be concentrated within subplans and minimized across subplans
3. Insert synchronization (typically communication) actions into subplans
4. Allocate subplans to agents using task-sharing
5. Initiate plan execution, and monitor progress

Distributed Planning for Distributed Plans

- *Most challenging* distributed planning is that both the planning process and its results are *distributed*

Research issues:

- Each problem solver (i.e. agent) has own objectives, how to make them work together in a coordinated fashion?
- How to work well with dynamic changes in uncertain environments?
- How to improve communication decisions so as to achieve a more global view for coordination?
- How to better coordinate agent activities so as to avoid harmful interactions and promote helpful interactions?

Distributed Planning for Distributed Plans

- *Most challenging* distributed planning is that both the planning process and its results are *distributed*
- Representative models/systems:
  - *CPEF*: Continuous planning and execution framework through cooperative interactions (Myers)
  - *DSIPE*: Constraint-based planner for plan merging (desJardins & Wolverton)
  - *PGP*: Partial global planning (Durfee, 1988, 1991, 1996)
  - *MMDP*: Decision-theoretic planning that extends multi-agent MDPs (Boutilier)
  - *CONSA*: Argumentation-based negotiation for team coordination (Tambe & Jung)

Partial Global Planning: Main Principle

Main Principle:

- Planning is *partial*

  It is not *feasible* to generate a plan for the entire problem

- Planning is *global*

  Agents form *non-local* plans by exchanging local plans and cooperating to achieve a non-local view of problem solving

Implications:

- An agent not necessarily need a *completely global* view in order to improve coordination
- An agent does need a *sufficiently global* view to recognize how changes to local plans could improve coordination

Partial Global Planning: General Mechanism

- In partial global planning (*PGP*): planning and coordination are *interleaved* with *three* iterated stages:
  1. Each agent decides what its own *goals* are, and generates *tentative local plans* to achieve them
  2. Agents *exchange information* to determine where plans and goals *interact*
  3. Agents *alter* local plans to *better coordinate* their own activities

Initially, an agent's partial global plans (*PGPs*) correspond only to its *local plans*. As information from other agents arrives, it builds larger and *more complete PGPs*

Distributed Vehicle Monitoring Application

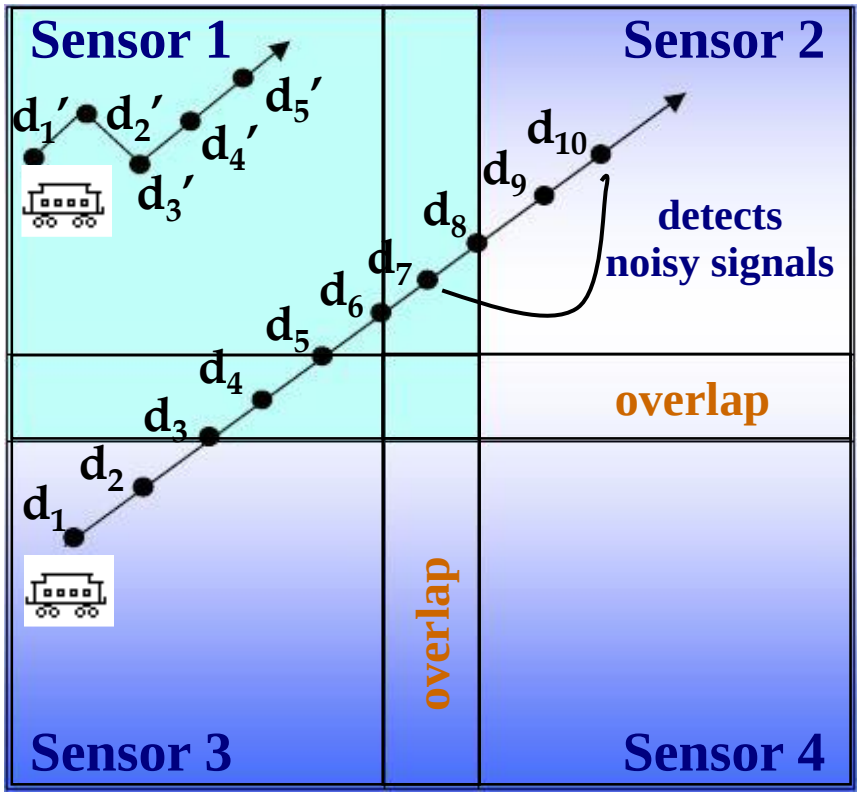
- Natural for distributed problem solving
  - Large amounts of geographically distributed incoming data
- Information is incrementally aggregated to generate map of vehicle movement

Each problem solving node:

- Responsible for only a local portion of the overall area
- Try to combine sensor data into tracks
- Plan to generate tracks

Node 1:

- Input data: (***d*$_1$'-*d*$_5$'**) and (***d*$_3$-*d*$_8$**)
- Local plans: ***plan*$_{11}$** and ***plan*$_{12}$**



How Nodes Work Together

- Communicate local plans to form multi-agent plans for coordination
  - *Nodes-1*, *2* and *3* work on the same larger goal and overlap
- Work independently and exchange partial results
  - *Node-2*'s sensor is faulty; *Node-1*'s result can help disambiguate noisy data
- Negotiate about task assignments in the network
  - *Node-1* is overloaded; *Node-4* has no tasks
- Coordinating node converges on *global solution*

➢ Partial global planning as a general *framework* to encompass these kinds of cooperation with different styles

Types of Plans in Partial Global Planning

Local plans:

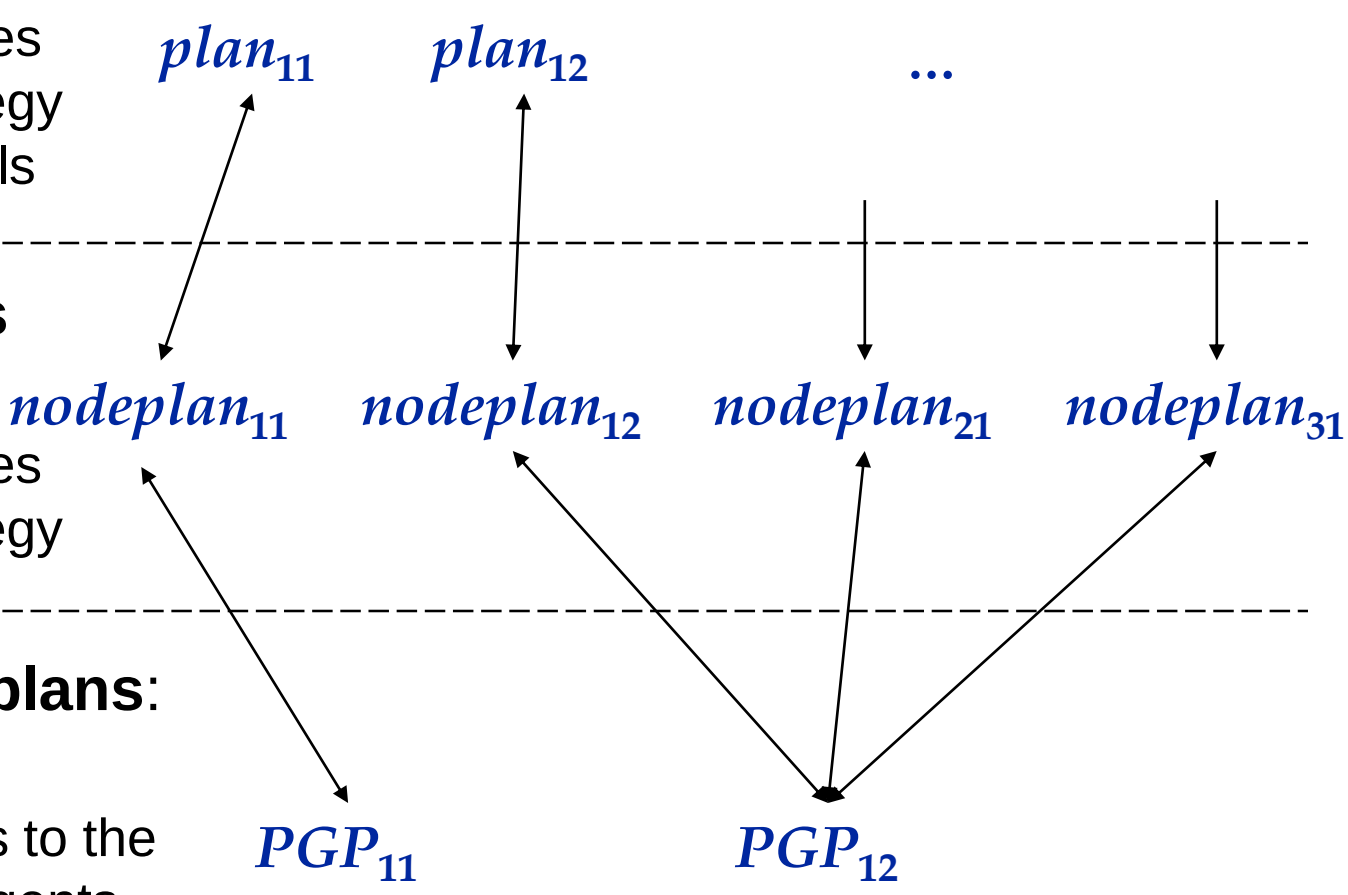
- A set of objectives
- Long-term strategy
- Short-term details

Abstract plans for node:

- A set of objectives
- Long-term strategy

Partial global plans:

- The larger goal
- A set of pointers to the plans of other agents



Partial Global Plans (PGPs)

- A *meta-level structure* incorporating the actions and interactions of agents to *guide* cooperation process
- A *data structure* that *specifies* *which agent* to exchange information with and under *what conditions*
- Enable different *styles* of *multi-agent* cooperation

Attribute	Meaning
Larger goal（较大目标）	The *PGG* that the system is working toward
Activity map（动作图）	Representation of *activities* and activity *effects* of agents
Solution construction graph（解的结构图）	*How* agents ought to interact, *what* information ought to exchange and *when*

Partial Global Planning: Steps

PGP assumes that planning and coordination decisions are continually revisited and revised:

1. Task decomposition: PGP tasks are *inherently* decomposed
2. Local plan formulation: an agent must first realize what *goals* to achieve and what *actions* to take to achieve goals
3. Local plan abstraction: agents are designed only to reveal their major *plan steps* (i.e. *abstraction* level) to other agents
4. Communication: agents communicate about their *abstract local plans* guided by *meta-level* organization structure
5. Partial global plan construction: local plans associated with the *same* partial global goal can be integrated into a *partial global plan* (at abstract plan step level) for individual agents

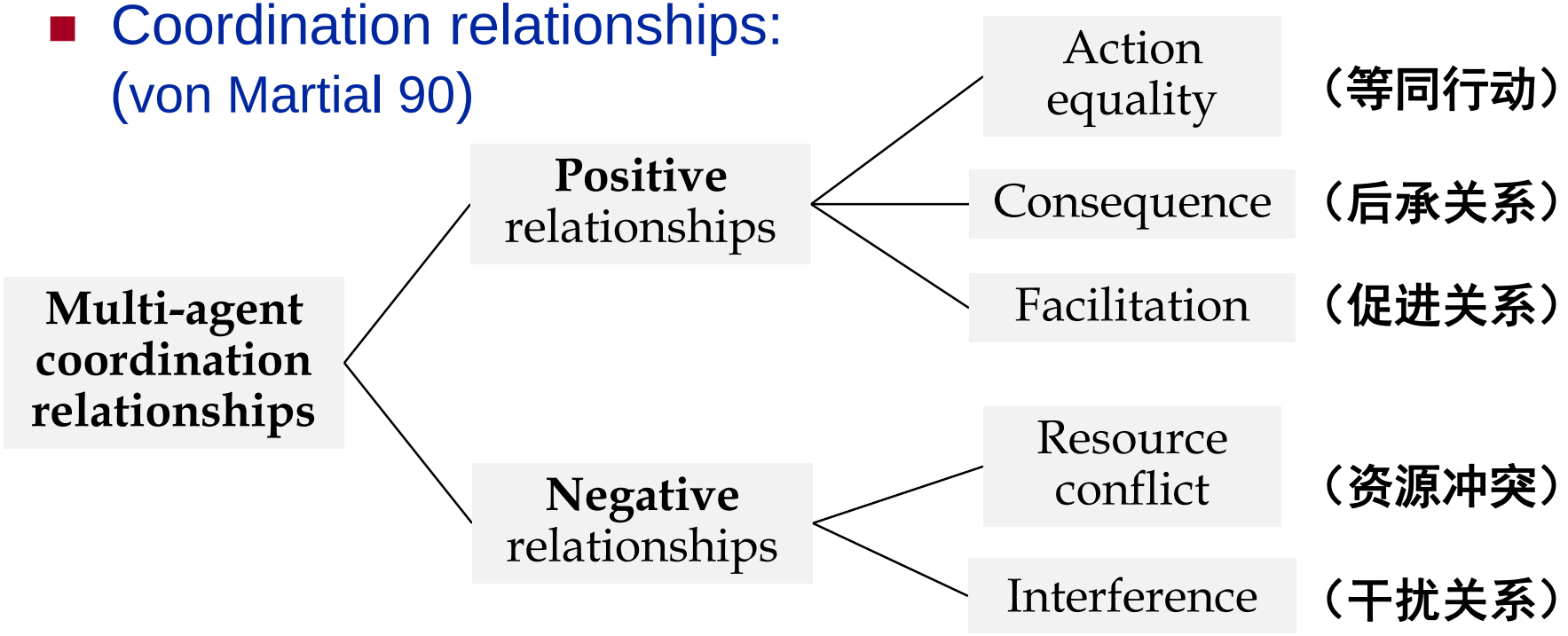
Partial Global Planning: Steps

6. Partial global plan modification: an agent with the constructed partial global plan can identify *opportunities* for improved *coordination*, by modifying certain plan steps
7. Communication planning: with a better coordinated plan, an agent can plan for *interactions*, by determining *when* and with *whom* to communicate the results of tasks
8. Acting on partial global plans: partial global plans are translated back to *local* level so that they can be *carried out*

Cycle repeats as local plans change or new plans from other agents arrive. At any given time, an agent acts based on its most recently updated local plan, but that plan might still be in the process of being coordinated

What to Coordinate in Agent Activities

- Coordination relationships: (von Martial 90)

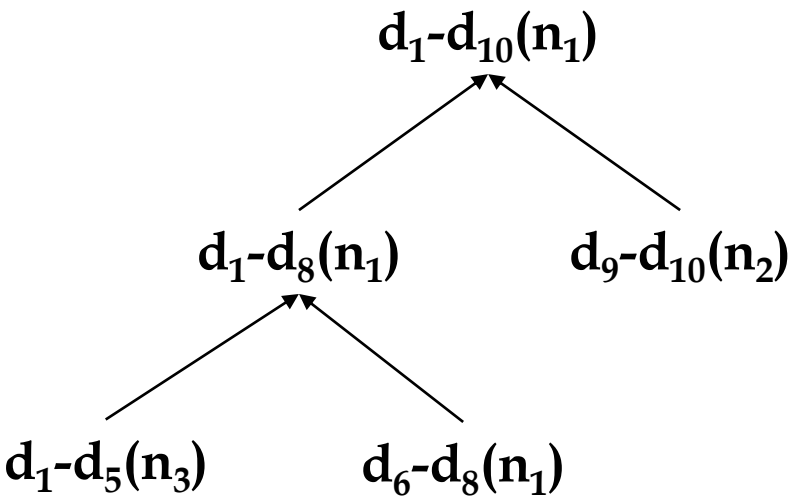


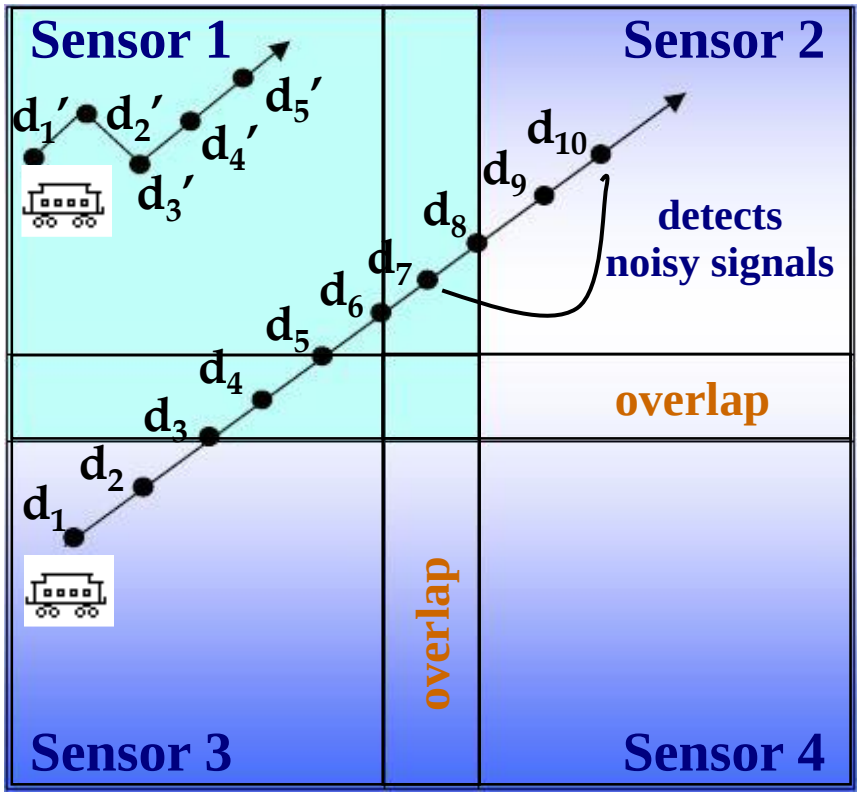
- **等同行动**：我们计划执行一个相同的行动
- **后承关系**：我计划的行动的执行导致你的某个(子)目标的实现
- **促进关系**：我计划的行动的执行帮助你的某个(子)目标的实现
- **干扰关系**：使计划执行的行动前提或结果不成立

Communication Coordination

- Selectively decide what partial results to communicate and with whom

Interaction strategy generated:





Trade-off:

- Send partial results in timely manner
- Send few, more complete results

Meta-Level Organization

- Centralized coordination

  where one node has authority over all others

- Hierarchical coordination

  where some middle management levels lie between top and bottom nodes

- Lateral coordination

  where all nodes have equal authority

➢ Meta-level organization influences how partial global plans are built and nodes' freedom to change their local plan, allowing various forms of *autocracy* and *democracy*, *obedience* and *insubordination*

More than Coordination

- PGP for task reallocation and *contracting*
  - Agents exchanging their abstract local plans can detect whether they are *overburdened*
  - Generate and propose *partial global plans* that represent candidate agents taking over some of its tasks
  - Establish a *contracting* relationship among agents
  - A recipient can *accept* or *counter-propose* by returning a *modified* partial global plan
  - *Negotiations* lead to task reallocation among agents
- Even applicable to more *centralized* tasks

Partial Global Planning: Features

- Focus on *dynamically* revising plans in *uncertain* world
  - In contrast to optimizing plans for static and predictable environments
- Different cooperation *styles* within one *unified* framework
- Can accommodate *task/result sharing* and *organization*
- *No* assumptions of the *global view* for agents
- Can work on *multiple plans* and goals

Limitations:

- *Inappropriate* for domains such as air-traffic control
  - *Guarantees* about coordination must be made *prior to* any execution

PGP: Applications and Extensions

- Work well for a number of task domains:
  - Distributed vehicle monitoring (Lesser & Corkill 83)
  - Distributed sensor network (Durfee & Lesser 91)
  - Cooperative information gathering (Decker et al 95)
  - Cooperative robotics (Clement & Durfee 99)

  … …

- Extensions
  - Generalized PGP (GPGP, Decker & Lesser 94)
  - Shared Activities (SHAC, Clement & Barrett 03)

Outline

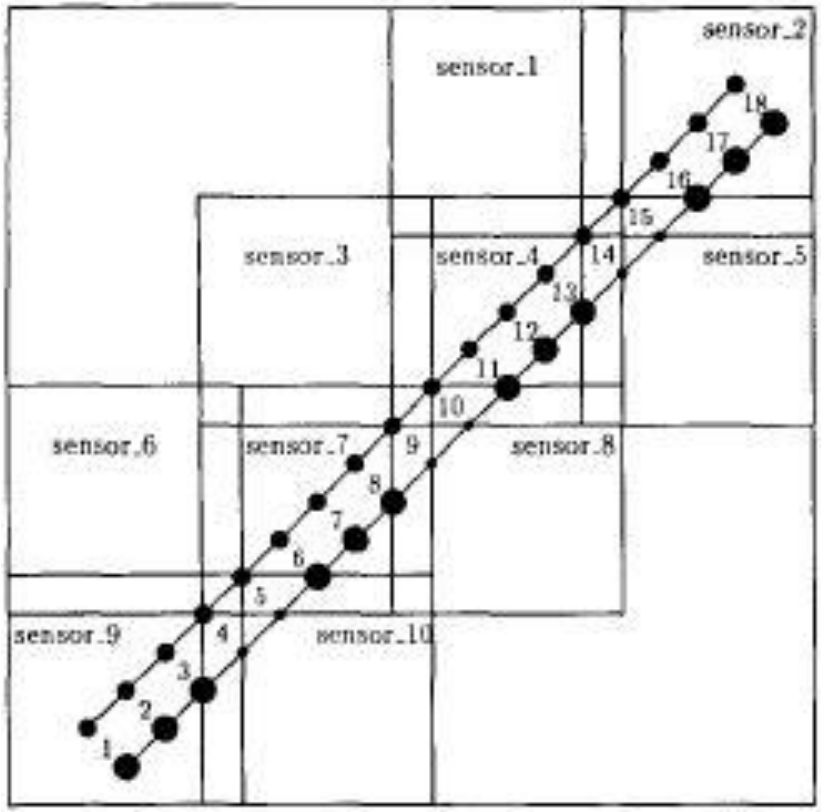
- Task sharing the contact net
- Result sharing in blackboard systems
- Distributed planning via partial global plans
- ➢ Distributed vehicle monitoring (DVM)

The DVM Testbed (Lesser & Corkill)

- Distributed vehicle monitoring testbed (DVMT)

Input sensor data:

- As vehicles move among the sensors, information about a signal's approximate *location*, *frequency class*, and *strength* is supplied at *discrete times* to corresponding nodes

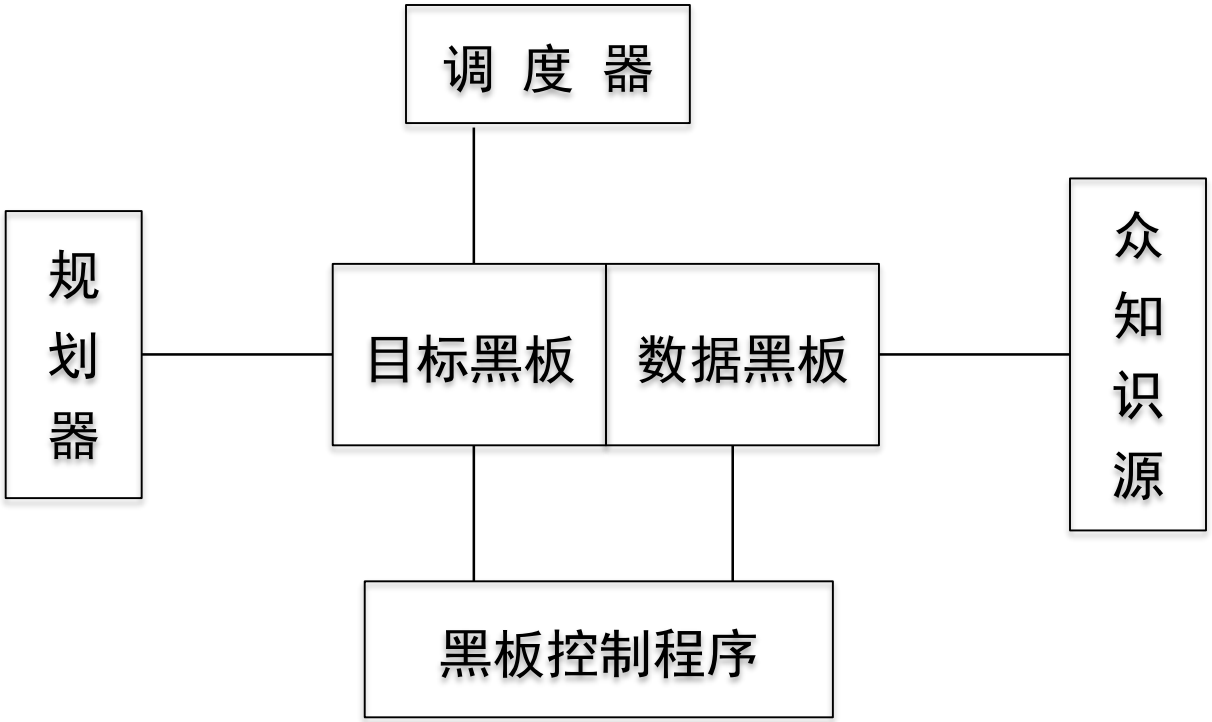


Node Architecture in DVMT

- Each node is an extended Hearsay-II system based on the *blackboard architecture*
  - with KS's appropriate for vehicle monitoring
  - capable of solving entire problem were it given all the data and used all of its knowledge
- Each node also has several extensions:
  - communication KS's
  - a goal blackboard
  - a planning module
  - meta-level blackboard control

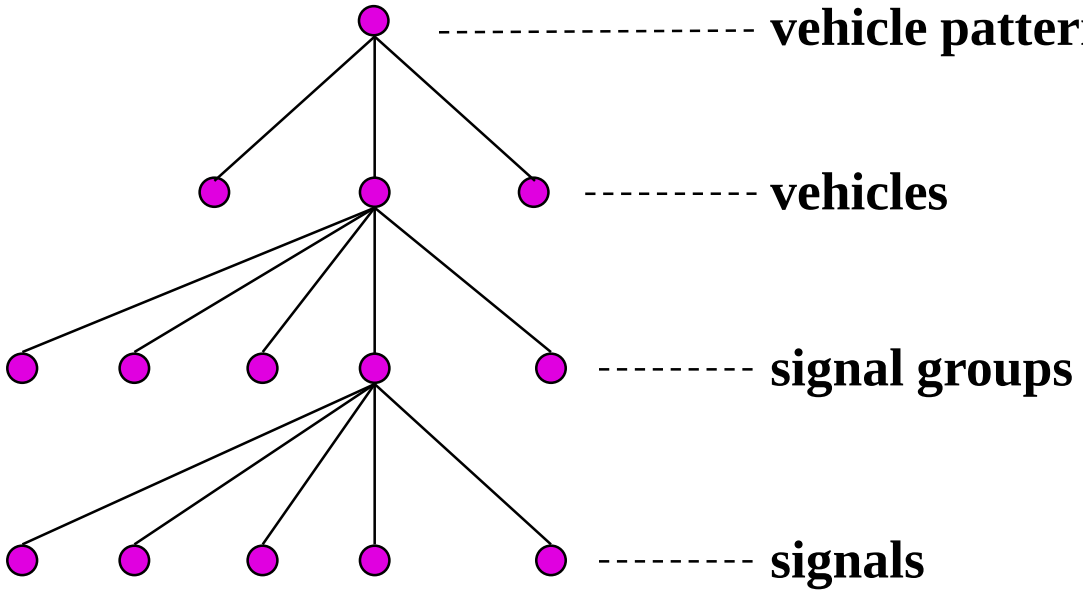
DVM系统的双黑板结构

- 通过扩充黑板上的信息内容，DVM系统的一个节点上的双黑板结构不仅包含数据黑板，还包含用于求解问题的目标黑板



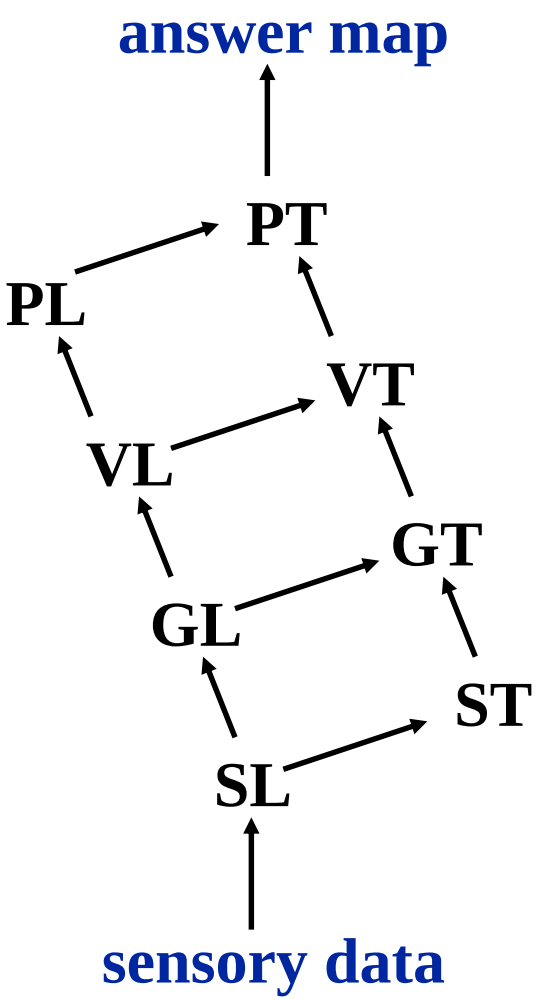
Task Processing Levels

- The data blackboard is partitioned into *4* abstraction levels:



- Each of these 4 levels is further divided into *two* levels:
  - One with *location hypotheses* (representing a single event at a particular time frame)
  - One with *track hypotheses* (representing a connected sequence of events over adjacent time frames)

Blackboard Levels in DVM



PT: ***pattern track***
PL: ***pattern location***
VT: ***vehicle track***
VL: ***vehicle location***
GT: ***group track***
GL: ***group location***
ST: ***signal track***
SL: ***signal location***

Knowledge Sources (KS's) in DVM

- Knowledge sources communicate through the shared blackboard to process data and incrementally construct interpretations, such as:
  - Grouping related signals together
  - Matching signal-groups to vehicle-types
  - Composing together sequences of partial interpretations into vehicle tracks

  Until overall interpretations are generated (i.e. answer map)

- There are six problem solving activities performed by KS's:

  Location synthesis, Track synthesis, Track formation, Track extension, Location-to-track joining, Track merging

Goal Processing

- Goal-directed control added to the pure *data-directed control* of Hearsay-II, through the use of a *goal blackboard* and a *planner*:
  - Goal blackboard: basic data units are goals, each representing an intention to create or extend a hypothesis on the data blackboard
  - Created by the blackboard monitor in response to changes on the data blackboard, or received from another node
  - Can bias the node toward developing the solution in a particular way

Planner and Scheduler

- Planner

  Respond to the insertion of *goals* on the goal blackboard by developing plans for their achievement and instantiating knowledge sources to carry out those plans

- Scheduler

  Use the *relationships* between the knowledge source instantiations and the goals on the goal blackboard to help decide how to use limited processing and communication resources of the node

Communication KS's

- Internode communication is added to the node architecture by the inclusion of communication KS's
- There are *six* types of communication KS's:
  - Hypothesis Send
  - Hypothesis Receive
  - Goal Send
  - Goal Help
  - Goal Receive
  - Goal Reply

Three Types of Plans

- Local plans
  - Information about plan's *objectives*, order of *major action steps*, how long each action will take, *detailed KS* list
- Node plans
  - *Abstract local plans* without the details of short-term actions
- PGPs

  Representation of how several nodes are working toward a partial global goals (e.g. several nodes track the same vehicle)
  - Information about the *PGG*, concurrent *node activities*, and a *solution construction graph* showing how partial solutions should be integrated together

Authority

- A higher-authority node can send a PGP to lower-authority ones to guide their actions in a certain way
- Two equal authority nodes can exchange PGPs to negotiate about (converge on) a consistent view of coordination
- A node receiving a node plan or a PGP considers the sending node's credibility when deciding how (or whether) to incorporate the new information into its own model

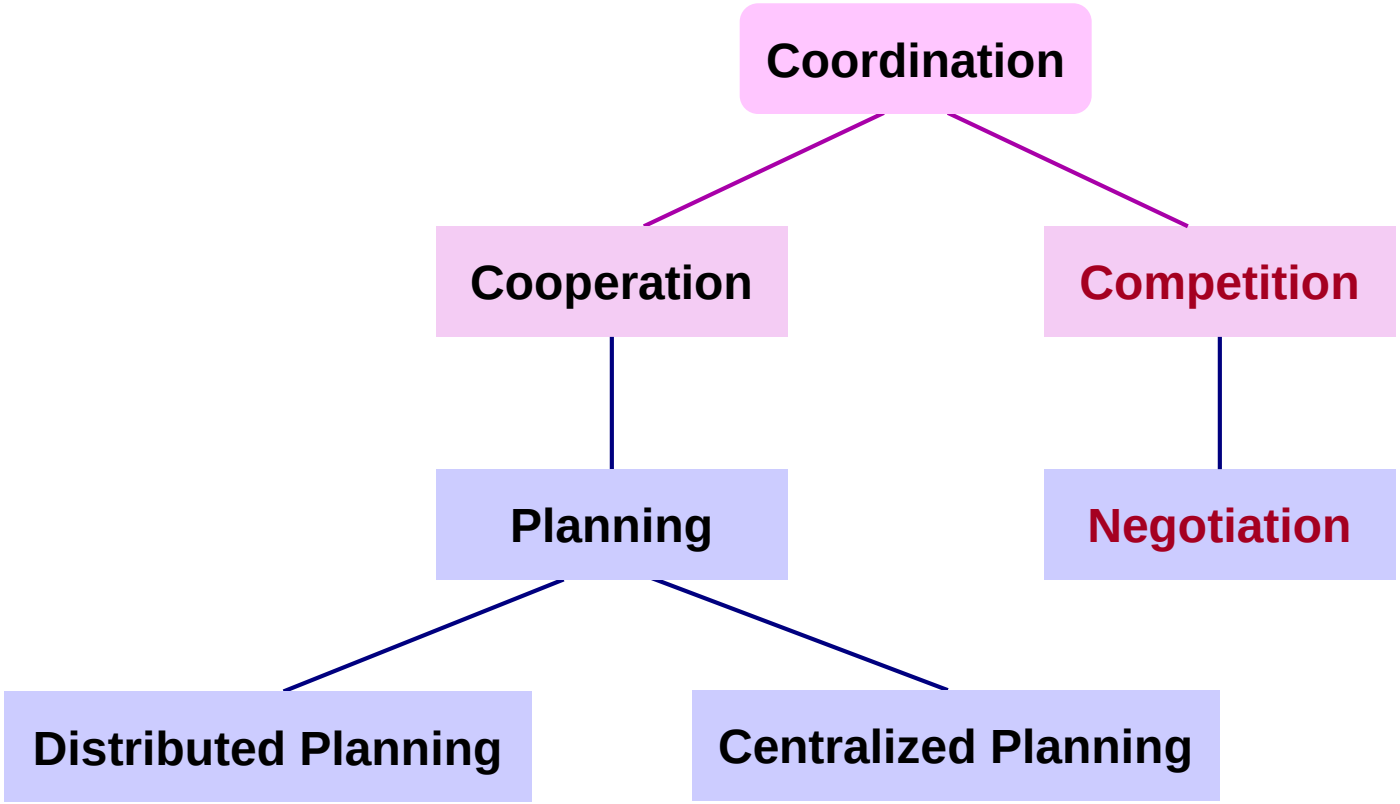
Experimental Results

- Larger networks involving *10* overlapping sensors
- Using nodes only plan locally as the baseline (E1)
- Meta-level organization:
  - Lateral (E2)
  - Central (E3)
  - Hierarchical (E4)

Expt.	Organization	Simulated sol. #time unit	Runtime	Comm. #message	#data structure
E1	local	49 / 65	583	216	1688
E2	lateral	42 / 57	429	415	1352
E3	central	42 / 50	208	363	1331
E4	hierarchical	41 / 51	161	403	1447

介绍：多智能体协商

- A taxonomy of some of the *different ways* in which agents can coordinate their behavior and activities



Game-Theoretic Approach to Negotiation

- Game-theoretic approaches to reaching agreement has distinct advantages to *prove* some desirable properties
  - You want to know “why should pay so much for this car”
  - “The agreement was the best for you” with a sequence of complex equations as an explanation

Disadvantages:

- Stances cannot be *justified*
  - When humans negotiate, they *justify* their negotiation *stances*
  - It is *hard* to understand how an agreement was reached in *GT*
- Preferences cannot be *changed*
  - Game theory tends to assume an agent’s utility function is *fixed*
  - However, our *preferences* do change when we negotiate

Argumentation Agent

- *PERSUADER* system (Sycara 89, 90)
  - *First* agent system using *argumentation-based negotiation*
  - Operate in *labour negotiation* domain with *3* agents:

    Labour union（工会）, Company（公司） and Mediator（仲裁者）
- Persuasive argumentation（劝说性辩论）to reach agreement
  - Iterative exchange of *proposals* and *counter-proposals* on *labor negotiation*

    Many issues: *wages*, *pensions*, *seniority*, *subcontracting*
  - *Reason about* another agent using own model of others
  - Find multiple ways to *influence* other agents' beliefs and behavior

Negotiation Process in *PERSUADER*

- Negotiation process consists of iteration of *3* main tasks:
  - The system (i.e. *mediator*) generates an initial *compromise proposal* and presents it to both agents
  - The *disagreeing party* generates a *counter-proposal* based on its evaluation of the proposal
  - The *mediator* makes a decision whether to *change* the initial *proposal* or attempt to change the disagreeing party's *position*

劝说性辩论

If both *accept* the proposal/modification, it is the final compromise

- Input: set of *conflicting goals* of agents and *dispute* context
- Output: an agreed settlement (i.e. *contract*) or an indication of *failure* (after a certain number of proposals)

Example: Generating an Argument

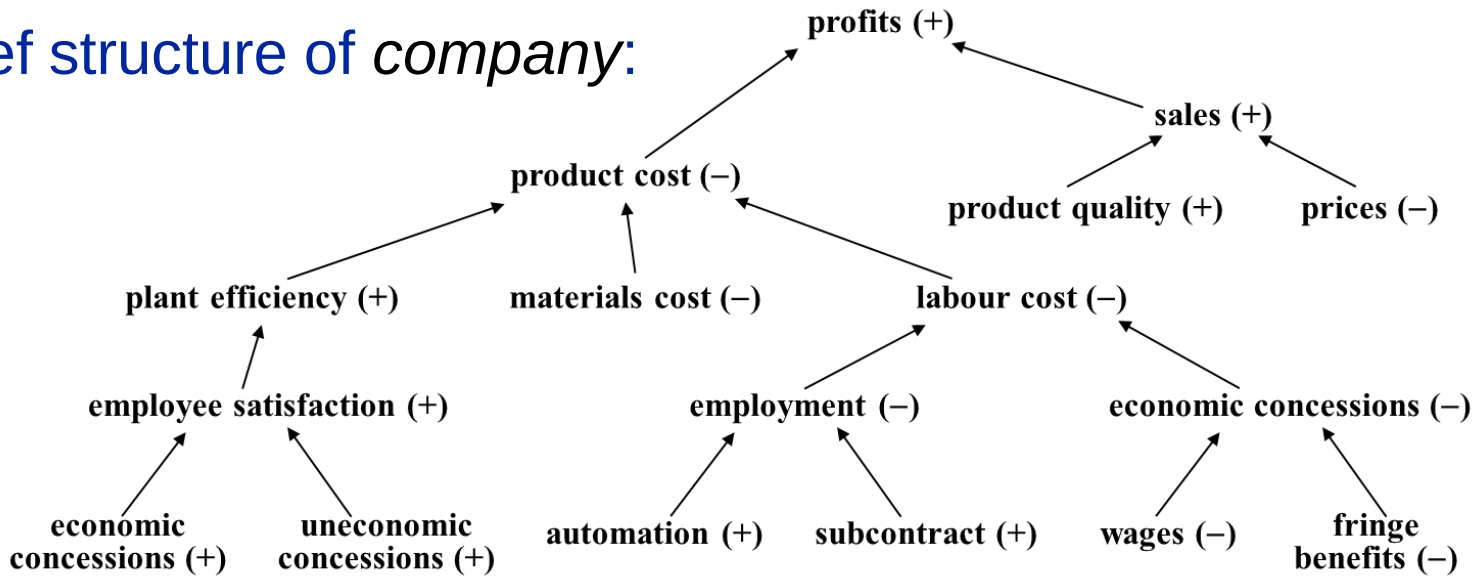
- Generating a *threatening* argument:

  If the company is forced to grant *higher wage increases*, then it will *decrease employment*.

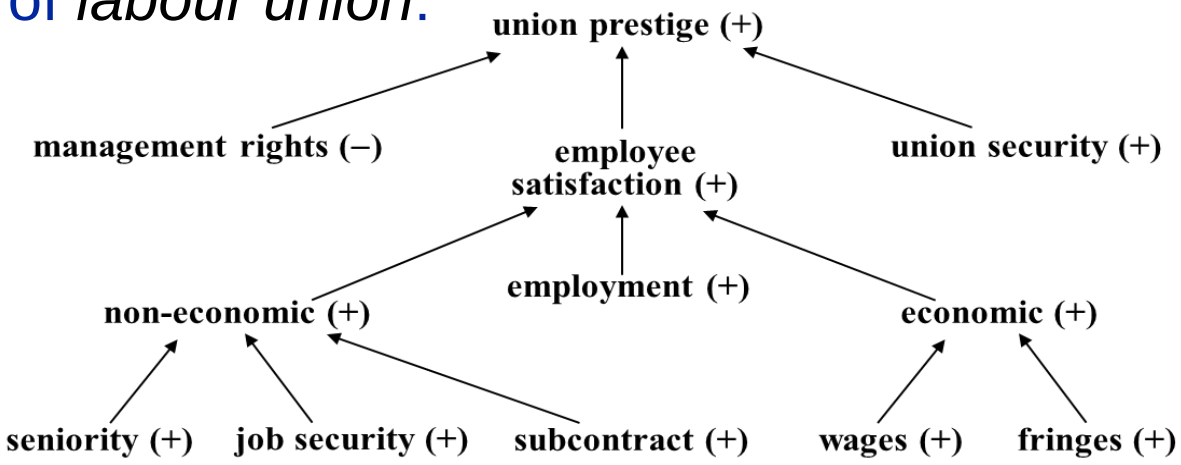
- Context:
  - *PERSUADER* has suggested a compromise
  - The *company* has agreed on a wage increase (i.e. the *highest* it can afford)
  - *Labour union* wants higher wage increases
- Argumentation goal:
  - Convince the *union* to accept the proposed increase and thus *abandon the goal* of higher wage increase

Belief Structure of Agents

- Partial belief structure of *company*:



- Partial belief structure of *labour union*:



Example: Generating an Argument

- An actual run of *PERSUADER* to generate the *argument*:

Importance of wage-goal is *6* for **union**

Search **company** goal-graph …

Increase in wage-goal by **company** will result in:
  increase in economic-concessions, labour-cost, production-cost
  decrease in profits

To compensate, **company** can:
  decrease fringe-benefits, decrease employment,
  increase plant-efficiency, increase sale

Only the following violates goals of **union**:
  decrease fringe-benefits, decrease employment

Importance of fringe-benefits is *4* for **union**

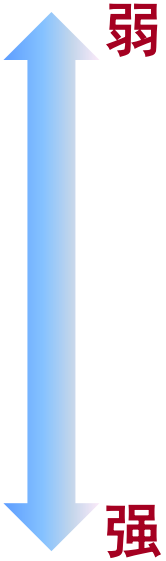
Importance of employment is *8* for **union**

Since importance of employment **>** importance of wage-goal, one possible argument found.

Argumentation Strategies

- Argumentations strategies are used to achieve argumentation goals

(1) Appeal to universal principle
(2) Appeal to a theme
(3) Appeal to authority
(4) Appeal to 'status quo'（现状）
(5) Appeal to 'minor standards'
(6) Appeal to 'prevailing practice'（普遍做法）
(7) Appeal to precedents as counter-examples
(8) Threats (Insisting current goal threatens a more important goal)



- In general, *PERSUADER* can generate *more than one* possible argument for a particular position

Other Argumentation Agents

- Extension of *PERSUADER* system

  Formalize some of the ideas in *PERSUADER* using logic and *implement* the formal version (Kraus et al 98)

- Logic-based model of argumentation

  Argumentation-based *negotiation* systems (Parson et al 98)

  Determine *winner* in argument dialogue (Amgoud 99)

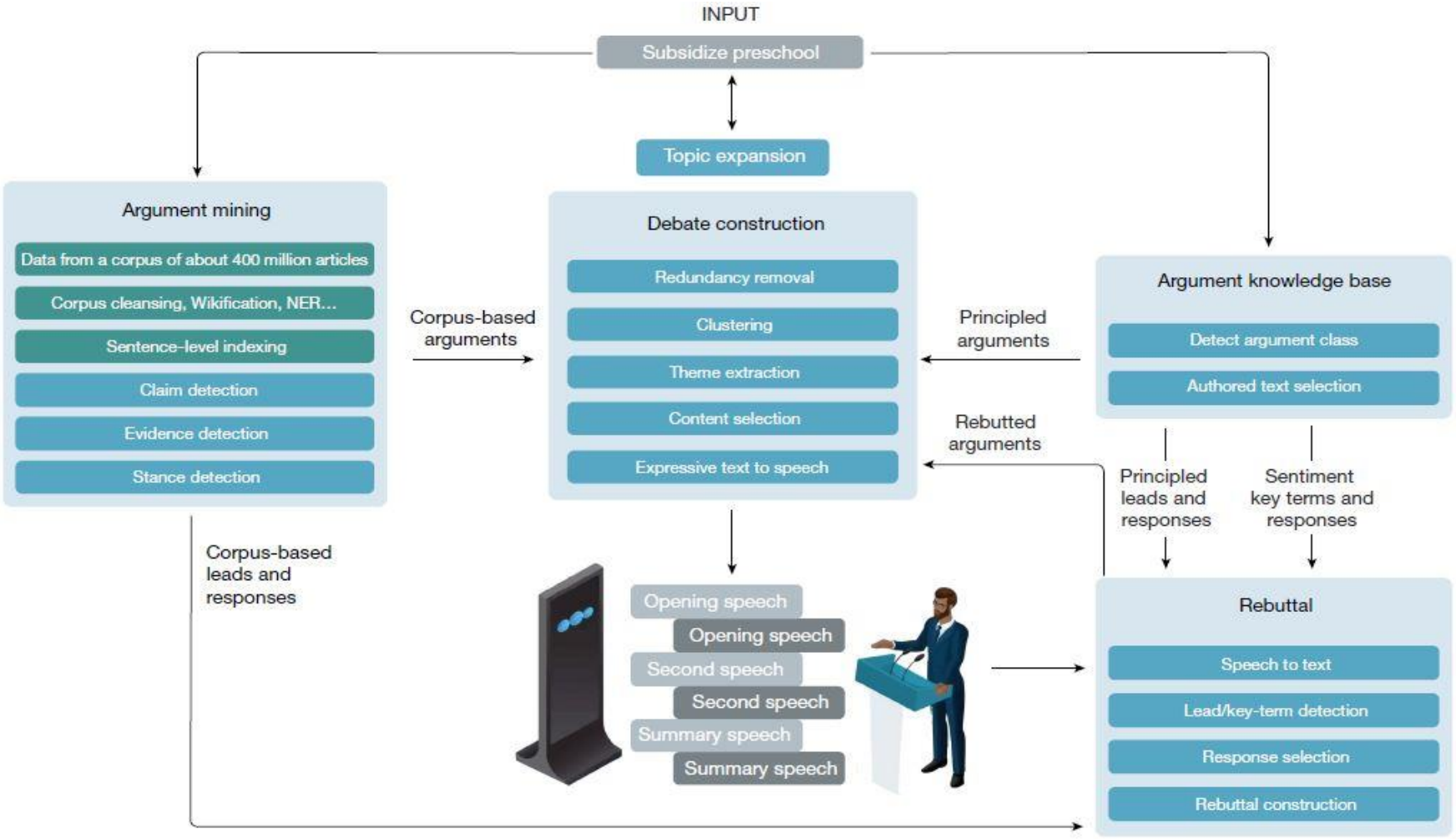
  Abstract argument systems (Vreeswijk & Prakken 2000)

- *Corpus-based* autonomous debating system (Nature 21)

  Argument and debate are fundamental capabilities of *human intelligence*, essential for a wide range of human activities, and common to *all* human societies

Project Debater (Nature 2021)

- Argument mining + Argument KB + Debate construction



Summary

- We have discussed how to get agents working together to do things
- There are different ways to have agents decide what to do, and make sure that their work is coordinated
- A typical multi-agent system may need a combination of these ideas

References

- R. G. Smith. *The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver*. IEEE Transactions on Computers, C-29(12):1104-1113, 1980
- R. D. Fennell and V. R. Lesser. *Parallelism in Artificial Intelligence Problem Solving: A Case Study of Hearsay II*. IEEE Transactions on Computers, C-26(2):98-111, 1977 (Also in *Readings in DAI*, pp. 106-119. Morgan Kaufmann, 1988)
- V. R. Lesser and D. D. Corkill. *The Distributed Vehicle Monitoring Testbed: A Tool for Investigating Distributed Problem Solving Networks*. AI Magazine, 4(3):15-33, 1983
- E. Durfee and V. R. Lesser. *Partial Global Planning: A Coordination Framework for Distributed Hypothesis Formation*. IEEE Transactions on Systems, Man, and Cybernetics, 21(5):1167-1183, 1991

References

- V. R. Lesser, K. Decker, T. Wagner, et al. *Evolution of the GPGP/ TAEMS Domain-Independent Coordination Framework*. Autonomous Agents and Multi-Agent Systems, 9(1-2):87-143, 2004
- K. P. Sycara. *Persuasive Argumentation in Negotiation*. Theory and Decision, 28:203-242, 1990
- M. Wooldridge. *An Introduction to Multiagent Systems (2nd Edition)*, *Chapter 8*. John Wiley & Sons, 2009
- E. Durfee and S. Zilberstein. *Multiagent Planning, Control, and Execution*. In: G. Weiss (Ed.). Multiagent Systems: Second Edition. MIT Press, 2013
- N. Slonim, Y. Bilu, C. Alzate, et al. *An Autonomous Debating System*. Nature, 591:379-384, 2021

End.