# Multiagent Planning:
# A Survey of Research and Applications

Brad Clement
Artificial Intelligence Group
Jet Propulsion Laboratory
California Institute of Technology
**brad.clement@jpl.nasa.gov**
**http://ai.jpl.nasa.gov/**

Keith Decker
Dept. of Computer and
Information Sciences
University of Delaware
**decker@cis.udel.edu**
**http://www.cis.udel.edu/~decker/**

## Outline

- What is multi-agent planning?
- Design Issues
- Applications
- Multi-agent planning problems and techniques
  - Planning for multiple agents
  - Planning by multiple agents
  - Coordinating before planning
  - Coordinating plans of multiple agents
  - Planning and coordinating
  - Distributed continual planning

Thanks to the following for contributions: Ed Durfee, Vic Lesser, Milind Tambe, Tom Wagner, Marie desJardins, Karen Myers, Carole Goble

# Orthodox Agent Definition

- An agent is a computer system that is capable of ***independent*** action on behalf of its user or owner
- ***An intelligent agent is a computer system capable of flexible autonomous action in some environment***
- By *flexible*, we mean:
  - **reactive** [change when environment changes]
  - **pro-active** [figuring out exactly how to achieve user goals, rather than being directly told)]
  - **social** [interact with others to do the above]

# Why Multiple Agents?

Five ongoing trends have marked the history of computing:

- *ubiquity*;
  - Continual reduced costs --> introduce processing power into places and devices that would have once been uneconomic
- *interconnection*;
  - Computer systems today no longer stand alone, but are networked into large distributed systems
- *intelligence*;
  - The complexity of tasks that we are capable of automating and delegating to computers has grown steadily
- *delegation*;
  - We are *giving control* to computers, even in safety critical tasks
- *human-orientation;*
  - movement away from machine-oriented views of programming toward concepts and metaphors that more closely reflect the way we ourselves understand the world

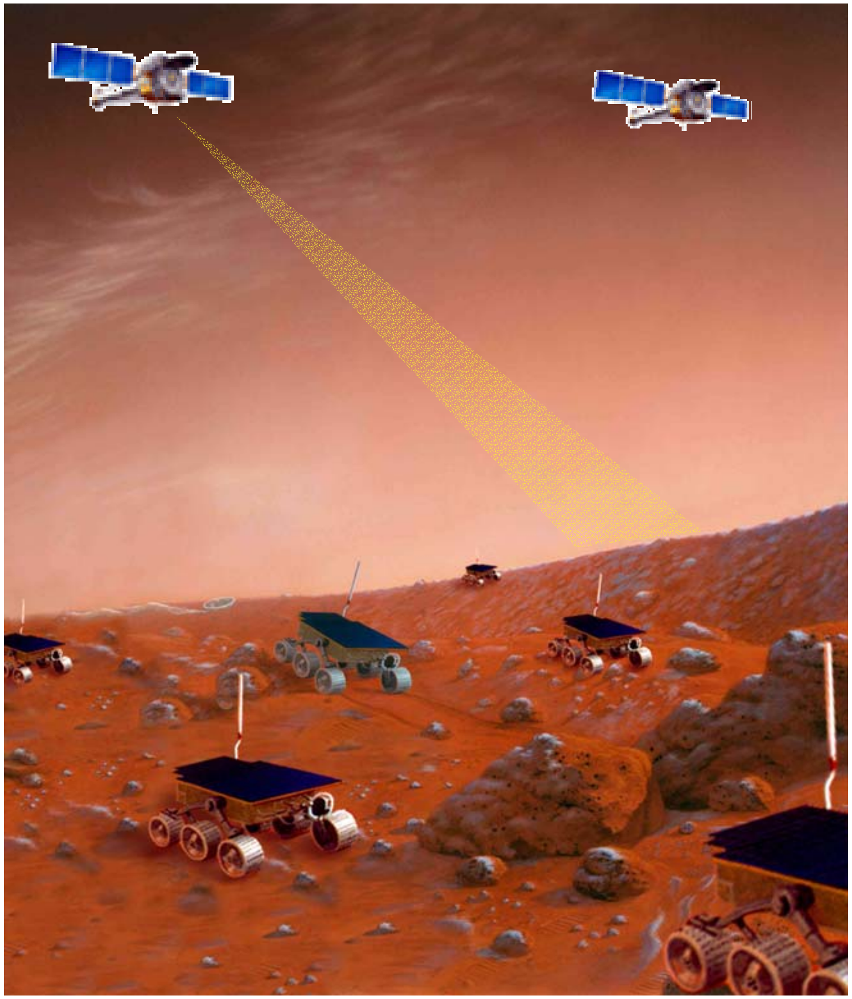
# Why Multiple Agents? cont.

- **Delegation** and **Intelligence** imply
  - The ability of computer systems to act independently
  - The ability of computer systems to act in a way that represents our best interests while interacting with other humans or systems
- **Interconnection** and **Distribution**, coupled with the need for systems to represent our best interests, imply
  - Systems that can cooperate and reach agreements (or even compete) with other systems that have different interests (much as we do with other people)

# Why multiple agents?

(Dias & Stentz, 2003)

- A single agent cannot perform some tasks alone
- A *robot* team can accomplish a given task more quickly
- A robot team can make effective use of specialists
- A robot team can localize themselves more efficiently
- A team generally provides a more robust solution
- A team can produce a wider variety of solutions
- Decision-making too costly or sensitive to centralize
- Multi-agent system already exists

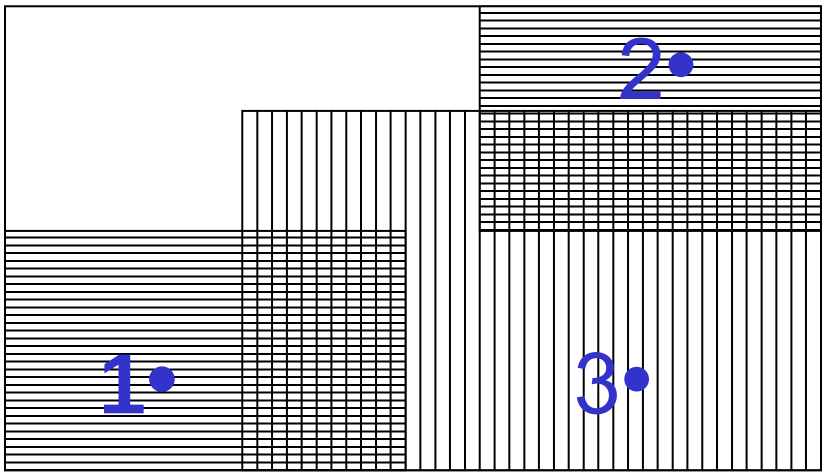


# Where do Agents in MAS Come From?

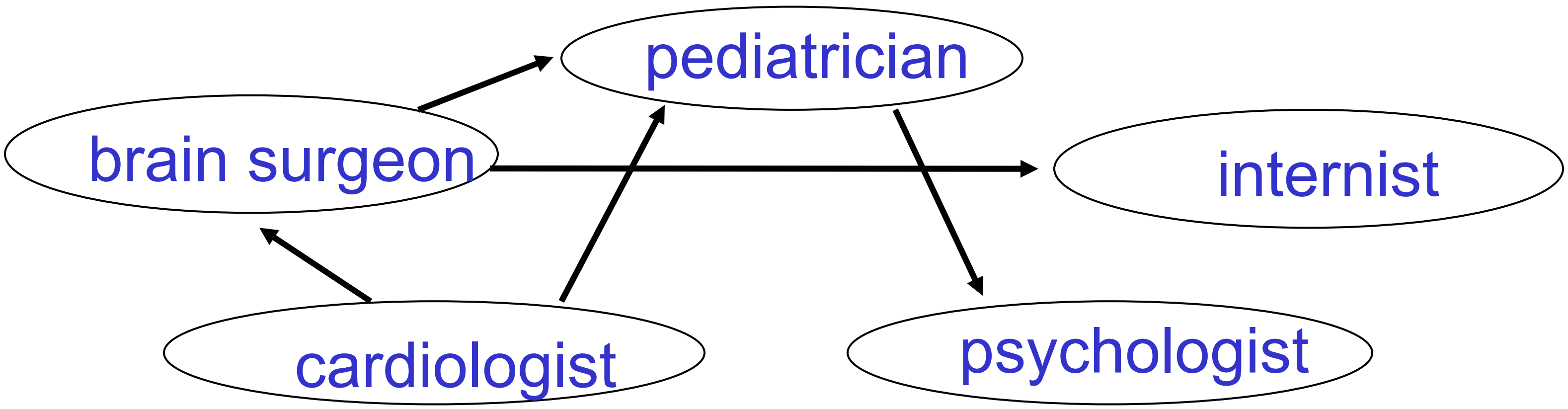
- Spatial, Functional or Temporal distribution of
  - information, expertise, resources, sensing and effecting
- Separate authority (lines of control) over resources
  - organizational imperatives
- Layered systems' architectures

# EXAMPLES

- *Spatial* decomposition (of information) distributed sensor network:




- *Functional* decomposition (of knowledge) group of experts:



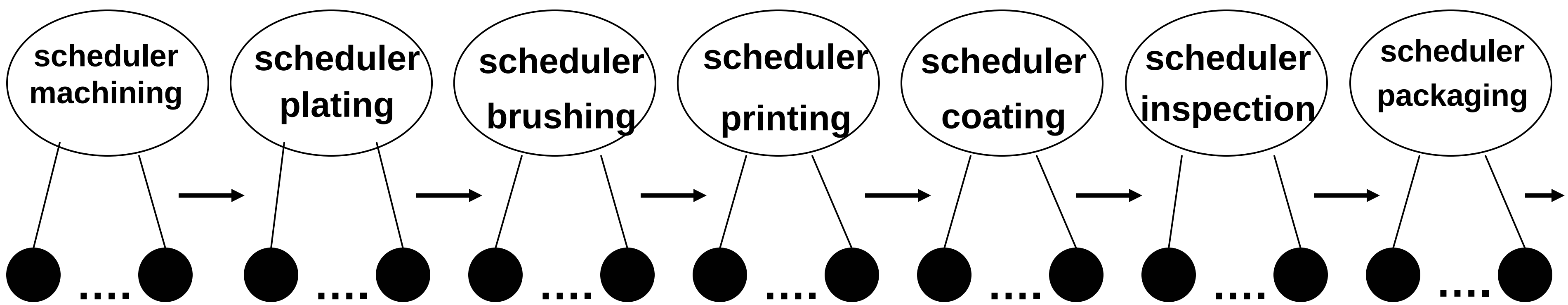
# *Temporal* Decomposition (of processing)

ORDERS

A Distributed scheduler for multi-layer, printed board line

MATERIALS

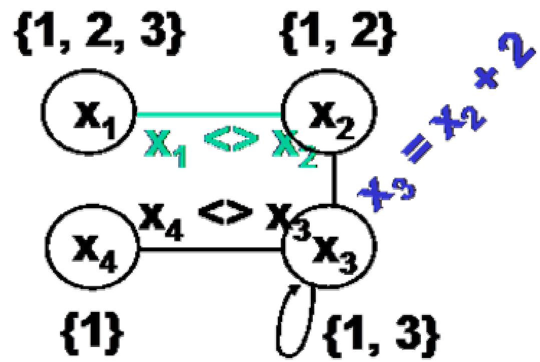
REQUEST FOR MATERIALS



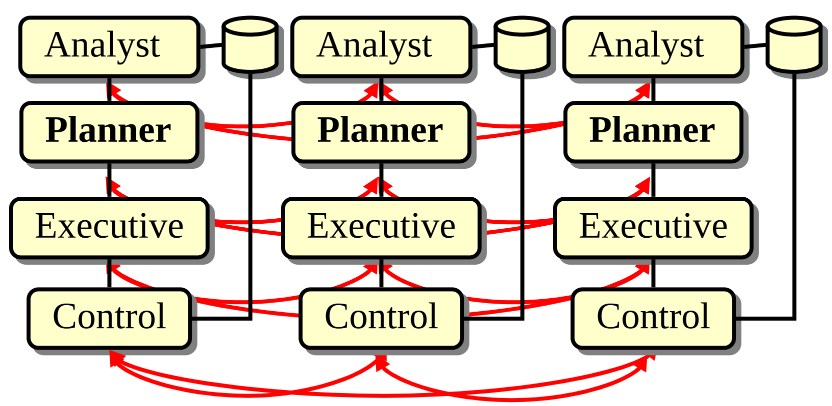

● = MACHINES

# Role of Multi-Agent Planning

- Multi-agent problem solving
  - Contract nets
  - Auctions
  - Game theory
  - Coalition formation
  - Distributed Constraint Satisfaction Problems (DCSP)
  - Distributed Constrained Heuristic Search (DCHS)
  - Complex systems
  - Multi-agent learning
  - **Multi-agent planning**



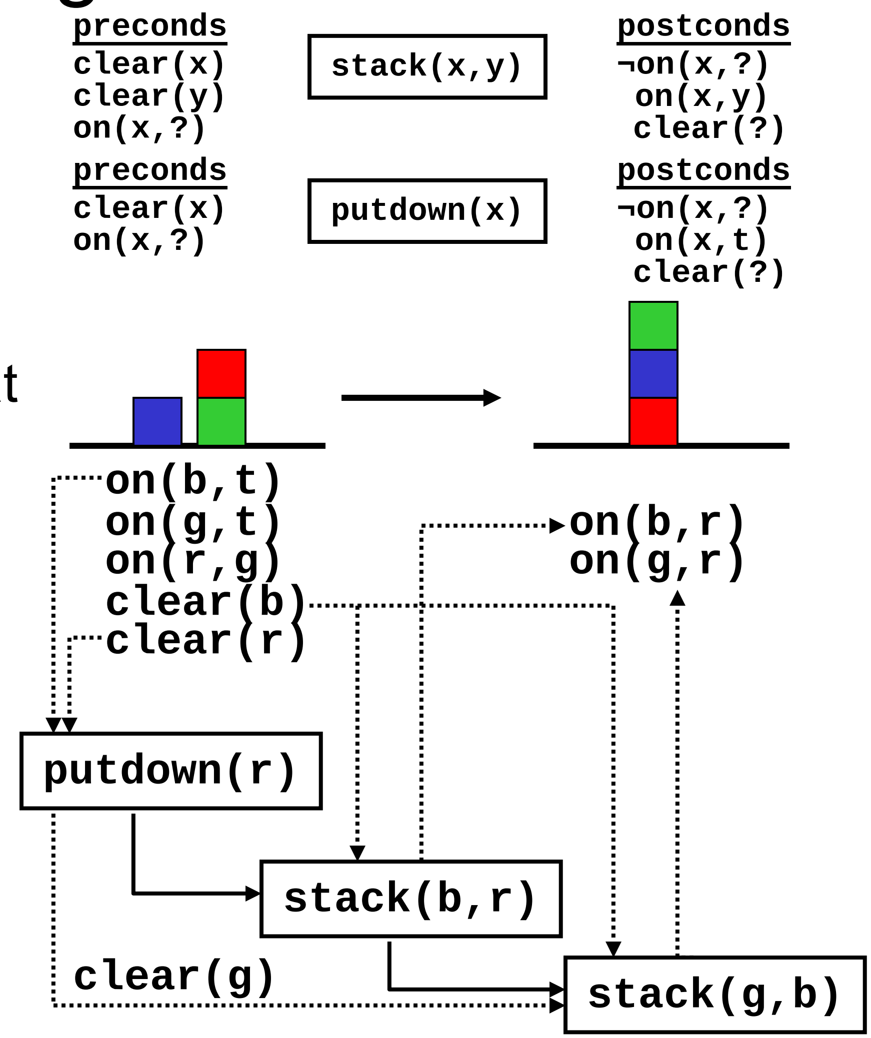

- Multi-agent system
  - Analysis, Meta-cognition
  - **Planning**
  - Execution
  - Control



# What is multi-agent planning?

## planning + agents

- Planning
  - near-term actions can effect subsequent ones in achieving longer-term goals
  - choose and order actions such that they lead from initial state to goals
- Multiple agents
  - Planning for multiple agents
  - Planning by multiple agents
  - Coordinating plans of multiple agents
  - Planning and coordinating
  - Distributed continual planning



# Outline

- What is multi-agent planning?
- *Design Issues*
- Applications
- Multi-agent planning problems and techniques

# Perspective on Coordination

- **Coordination: the act of managing interdependencies between activities**
- (Begs the question of what "*managing*" and "*interdependencies*" imply… )

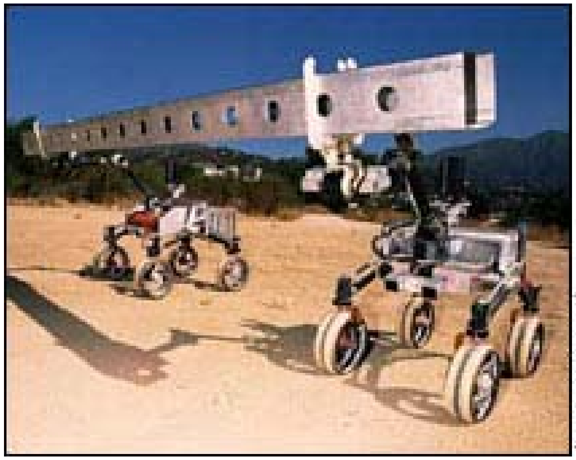
# Coordination: “Managing” Interdependencies

- Selecting goals/objectives/desires/intentions
- Planning to achieve these
- Scheduling actions within the resulting plans
  - Relative action ordering (”choreography”)
  - Absolute action placement (”synchronization”)

- Planning and Scheduling are on a continuum and will be considered together here

# Why coordinate? Interdependencies!

- Competing objectives (limited shared resources)
  - Shared parts and machines in factory
  - Battery power/energy
  - Market (goods, jobs)
- Shared objectives requiring joint actions
  - Carrying a beam
  - Joint sensing



(NASA/JPL)

# Coordination: Managing “Interdependencies”

- If there is a choice, then the particular action/agent chosen matters [SELECTION]
  - high quality, slow actions vs fast, lower quality approximations
  - alternative agents
- The order in which actions are carried out matters [CHOREOGRAPHY]
  - hard precedence constraints
  - soft facilitation opportunities
- The time at which actions are carried out matters [SYNCHRONIZATION]
  - hard or soft deadlines; earliest start times; etc.
  - time implies ordering across multiple agents

# Why is Coordination Hard?

- No global view
- Dynamically changing situation
- Uncertainty in the outcomes of actions
- Computational complexity of mapping problem (selection + scheduling)
- Scale in #agents and #tasks
- Deadlines / Time pressure
- Agent self-interest
- Non shared utility

# Coordinating Computational Actions

- Primary difficulties in CHOOSING and TEMPORALLY ORDERING actions because:
  - incomplete view of the problem
  - dynamically changing situation
  - uncertainty in the outcomes of actions

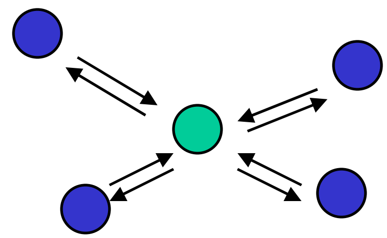
# Coordinating Computational Actions

- Overcome difficulties with Coordination Mechanisms
  - schedules, plans, timelines, appointments, commitments
    - Partial views, mostly static situation, often little action uncertainty
  - laws, rules, social behavioral norms
    - Ignore view, possible contingent decisions, reduce uncertainty
  - organizations, roles, negotiated order
    - Allow multiple views, abstract the situation, reduce uncertainty

# Coordinating Computational Actions

*Another view*

- Centralized Coordination Mechanisms
  - single locus of data/knowledge and decision-making/authority
  - PROS: easier to show optimality, implement, ignore concurrency issues, communicate only twice (gather problem info, issue results)
  - CONS: central point of failure, human organizational mismatch, difficulties in dynamic environments



- Decentralized Coordination Mechanisms
  - decentralized knowledge/data and decision-making
  - PROS: parallel computation, communication constraints (e.g. privacy), robustness, organizational fit, opportunistic, realistic
  - CONS: rarely optimal compared to centralization, concurrency complexity

- Reality: hybrids (e.g. centralized control of individual resources in a decentralized environment/context)

# Coordinating Computational Actions

*Yet another view*

- Static Coordination Mechanisms
  - designed by programmers at design-time
  - example: rules of the road
- Dynamic Coordination Mechanisms
  - "designed" by agents at run-time
  - parameterized static mechanisms
  - selection between static alternatives

# Coordinating Computational Actions

*Yet another view*

- Implicit Coordination Mechanisms
  - Altering/defining the environment so as to "solve" the coordination problems
  - e.g. Social Conventions/Laws
  - e.g. Organizations
  - e.g. Agent Modeling
  - e.g. Free Market Economics ("the invisible hand")
- Explicit Coordination Mechanisms
  - Agents explicitly "arguing" over who does what, and when
  - e.g. Representing & Exchaning Commitments
  - e.g. Distributed Planning
  - e.g. Distributed Scheduling
- Reality: Hybrids, "open and closed questions"

# Coordination vs. Coherent Action

*[Searle]*

- Implicit Coordination -/-> coherency
  - Robot 1 observes Robot 2 heading for Exit 2
  - Therefore, Robot 1 decides to use Exit 1
  - However, observation was misleading; Robot 2 also heads for Exit 1
- Coherent Action -/-> explicit coordination
  - Observe many people from all over the place running to a central tree (coherent action)
  - Context:
    - (explict coordination) Dancers in a ballet
    - (implicit coordination) People trying to avoid sudden rain in the park

# Coordinating Computational Actions *(summary)*

- Abstraction
  - Goals
  - Plans
  - Schedules
- Location
  - Centralized
  - Decentralized
- Learning
  - Static
  - Dynamic
- Structure
  - Implicit
  - Explicit

# Criteria for Multi-Agent Planning

1. computation costs
2. communication costs
   - number of messages
   - data volume (required bandwidth)
   - required latency
3. flexibility (commitment)
   - how much freedom do agents give each other
   - time
   - resources
   - choice of action
4. robustness – ability to succeed in changing environment
5. plan quality (could be a function 2,3,4)
6. scalability
   - number of agents?
   - size of problem input/output
   - interactions

# Outline

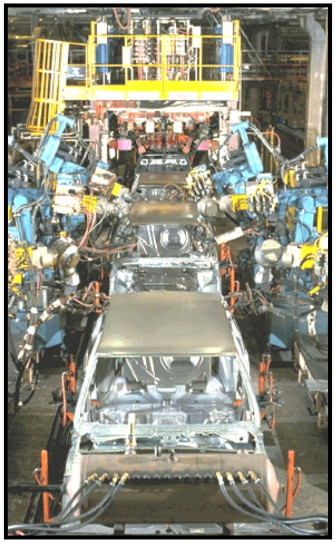
- What is multi-agent planning?
- Design Issues
- *Applications*
- Multi-agent planning problems and techniques

# Industry Applications

**Decentralize decision-making?**

- **competing objectives (self-interest)**
- **control is already distributed**
- **communication constraints/costs**
- **computation constraints**

- car assembly
- factory management
- workforce management

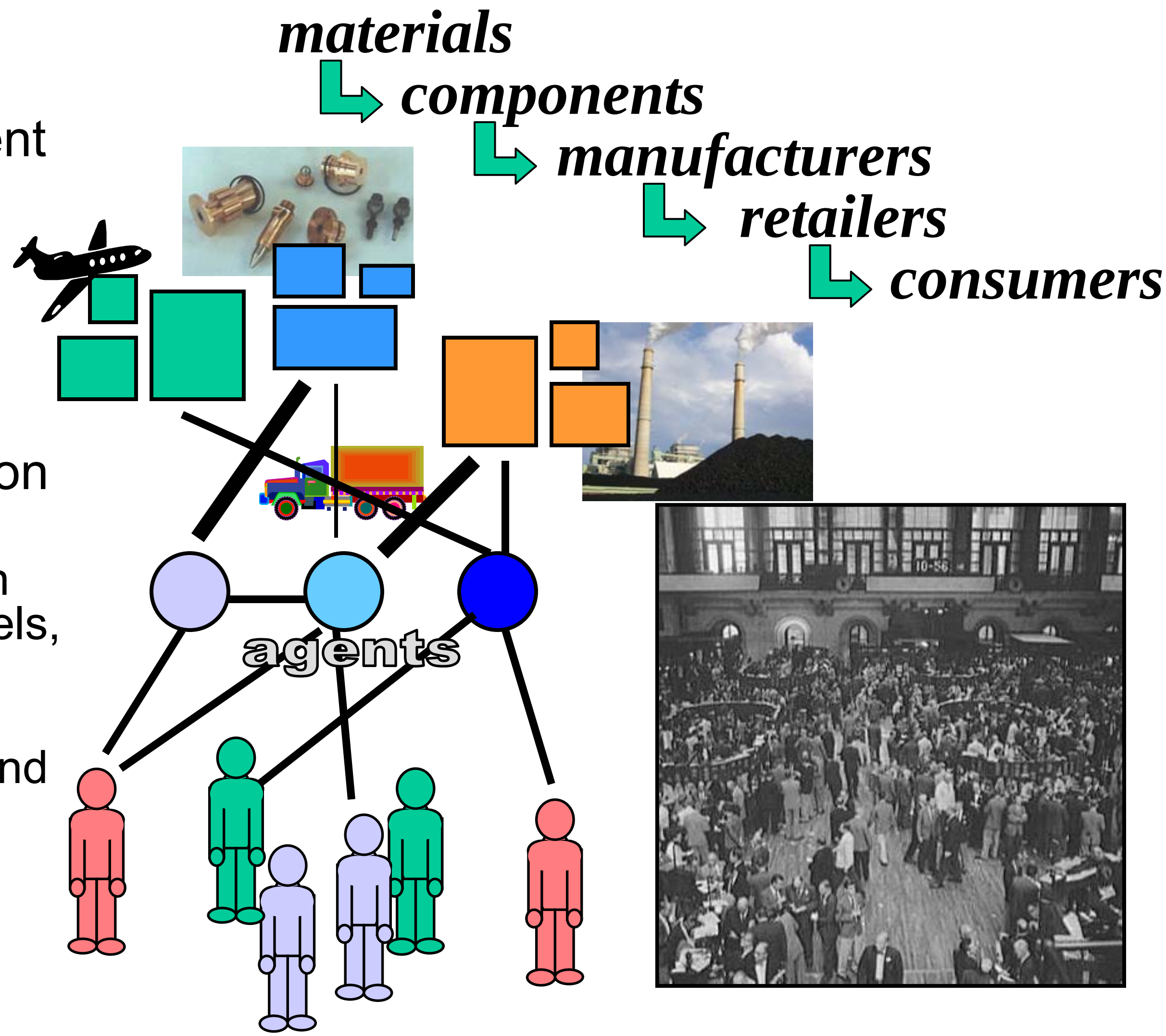


# Market Applications

**Decentralize decision-making?**

- **competing objectives (self-interest)**
- **control is already distributed**
- **communication constraints/costs**
- **computation constraints**

- Supply chain management (SCM)
  - inventory management
  - distribution logistics
  - buy/sell strategies
  - coalition formation
- Trading Agent Competition (TAC)
  - travel agents – between clients and airlines, hotels, and ticket offices
  - SCM – manufacturers buying from suppliers and selling to customers
  - auctions
- stock market?

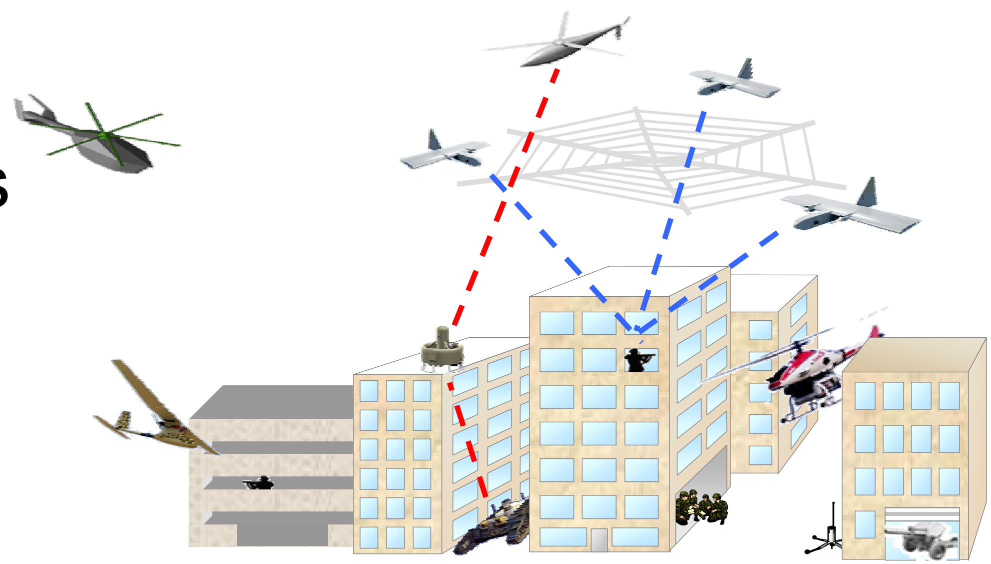


# Military Applications

**Decentralize decision-making?**

- **competing objectives (self-interest)**
- **control is already distributed**
- **communication constraints/costs**
- **computation constraints**

- distributed sensors
- unmanned vehicles
- troop/asset management
- submarine automation



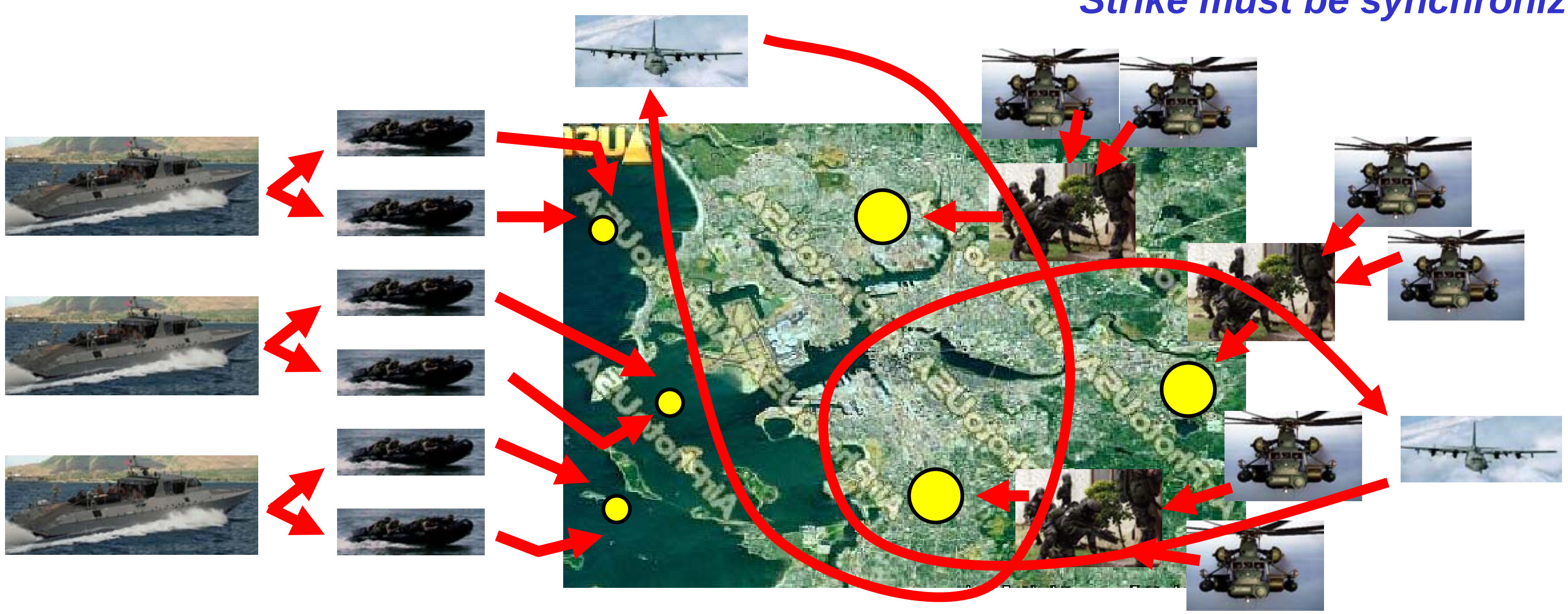
# Tactical Team Coordination: Effective Operations Require Units to Act in Concert....

Example: Hostage rescue

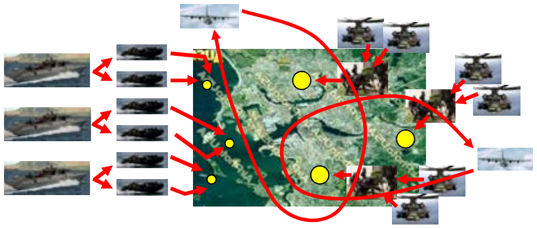
Intelligence indicates cells in three ships and three land targets

- Task force created and deployed:
  - Joint HQ.
  - Company of Special Forces (Army).
  - Platoon of SEALs (Navy).
  - Four small boats (Navy) w/rafts.
  - Troop helicopters (Air Force).
  - Support gunships (Air Force).
  - Civilian and support units.
- Mission: First SO & S&R teams. Then hit all targets simultaneously. H-Hour set to 0100 local time (7 hours before deadline).

***Strike must be synchronized.***



Tactical Team Coordination: Effective Operations Require Units to Act in Concert.... and to Adapt Continuously to Change



**Initial Plan** – formed *a priori* / offline

Specifies who should be doing what, when, with whom, etc. **Static!**

Deploy

**Change – Friction of War**

*Example: Engine failure aboard one Navy boats delays mission.*

**Change – Environment**

*Example: Weather conditions put support gunships into a wider flight path – will take longer to provide support.*

Affects

Affects

**Current Plan** – *constantly changing* in both small and large ways.

Affects

Affects

**Change – Command Decisions**

*Example: Intelligence indicates two of the ships will leave the harbor sooner than anticipated.*

**Change – Due to Enemy**

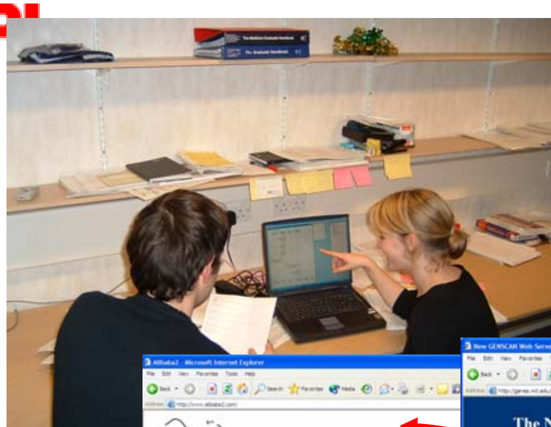
*Example: The cell in one ground target boards a pair of vans and heads out onto the highway.*

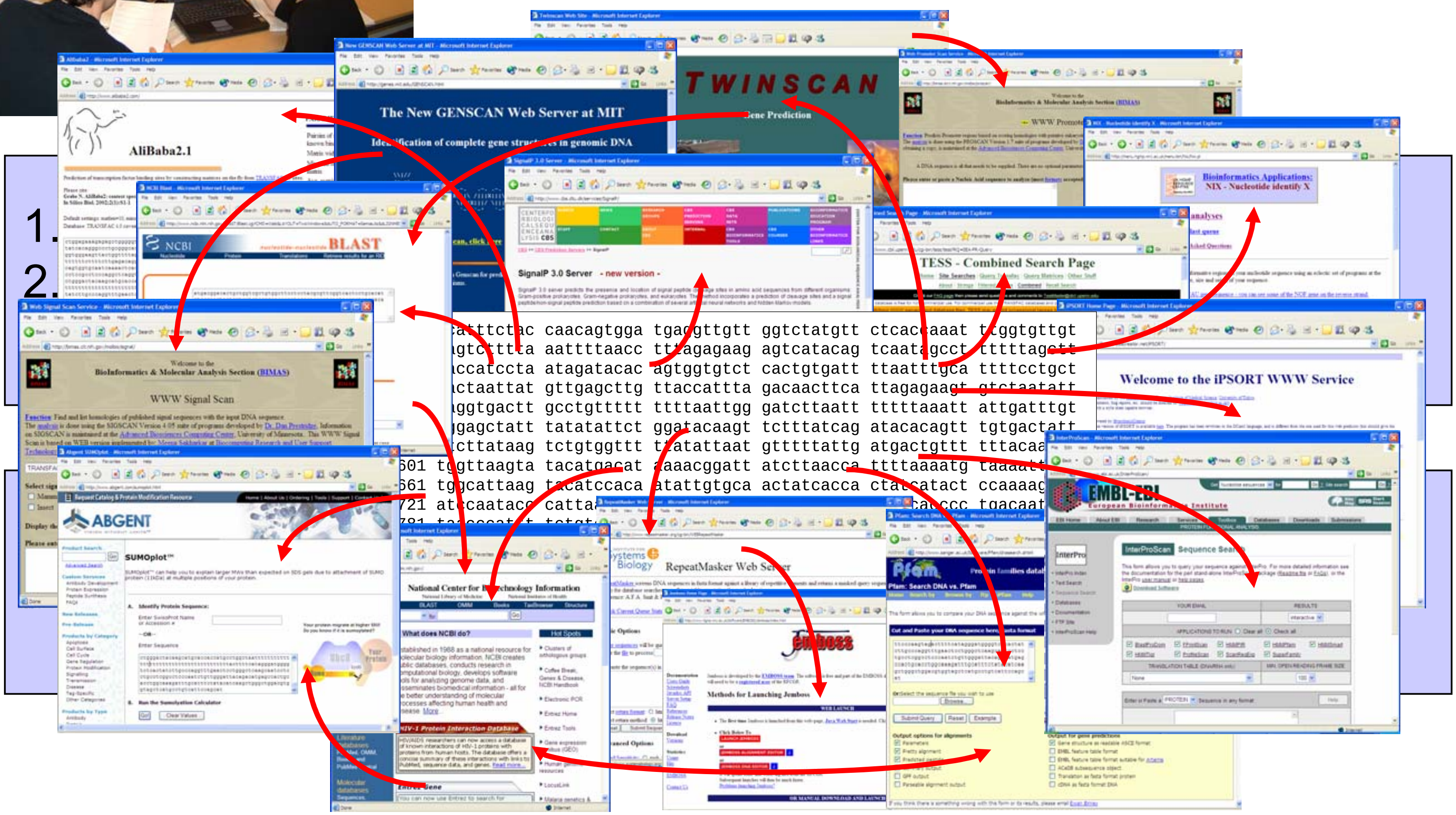
**Online Adaptation**

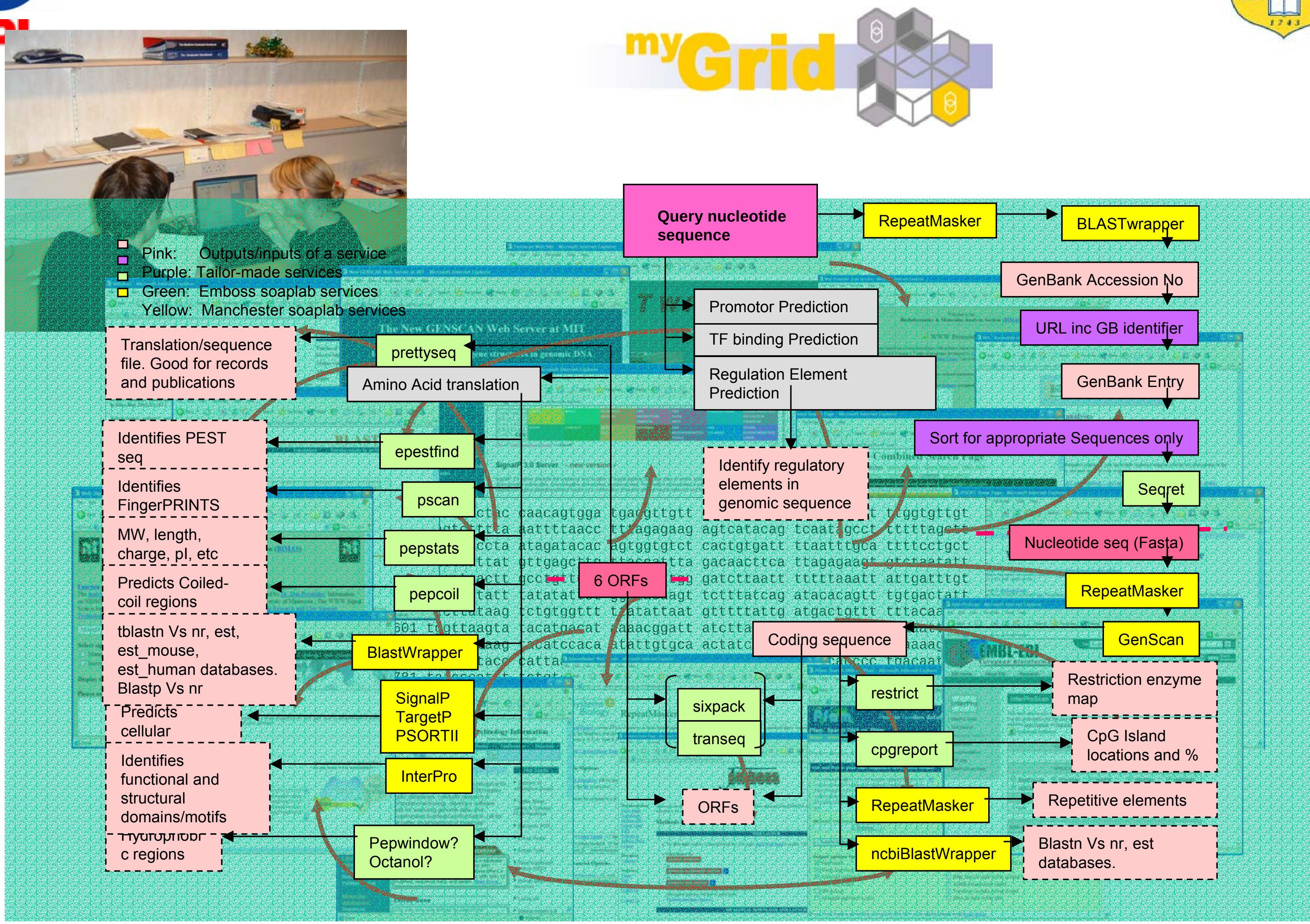
- Both at command and unit levels.
- Most require *coordination (not replanning)* – changes to task timing/allocation or contingency selection.
- Focus: what, when, who.

# E-Science Applications

- Distributed planning and scheduling of computational workflows
  - Examples:
    - MyGrid
    - BioMAS
- Planning for simulation

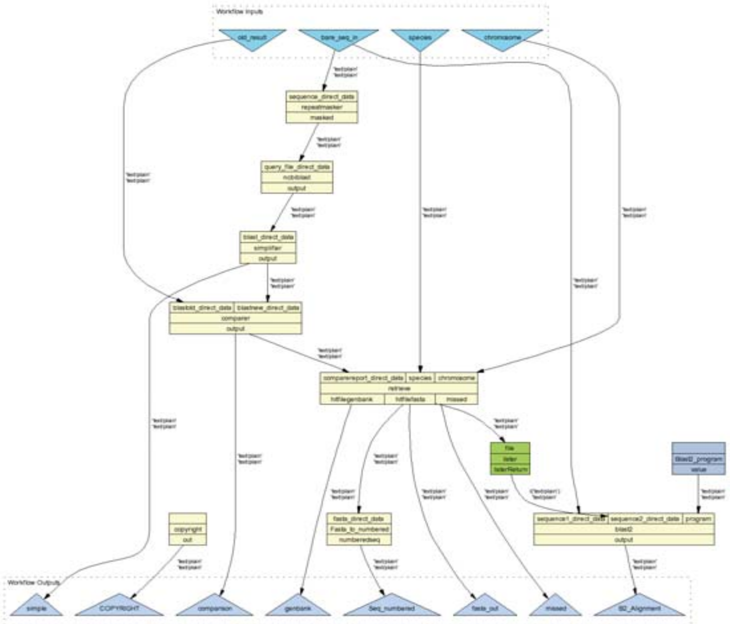



1.
2.
AliBaba2.1
The New GENSCAN Web Server at MIT
TWINSCAN
BLAST
SignalP 3.0 Server - new version -
TESS - Combined Search Page
Bioinformatics Applications: NIX - Nucleotide identify X
Welcome to the iPSORT WWW Service
WWW Signal Scan
ABGENT
SUMOplot
National Center for Biotechnology Information
RepeatMasker Web Server
EMBL-EBI
InterProScan Sequence Search
Methods for Launching Jemboss

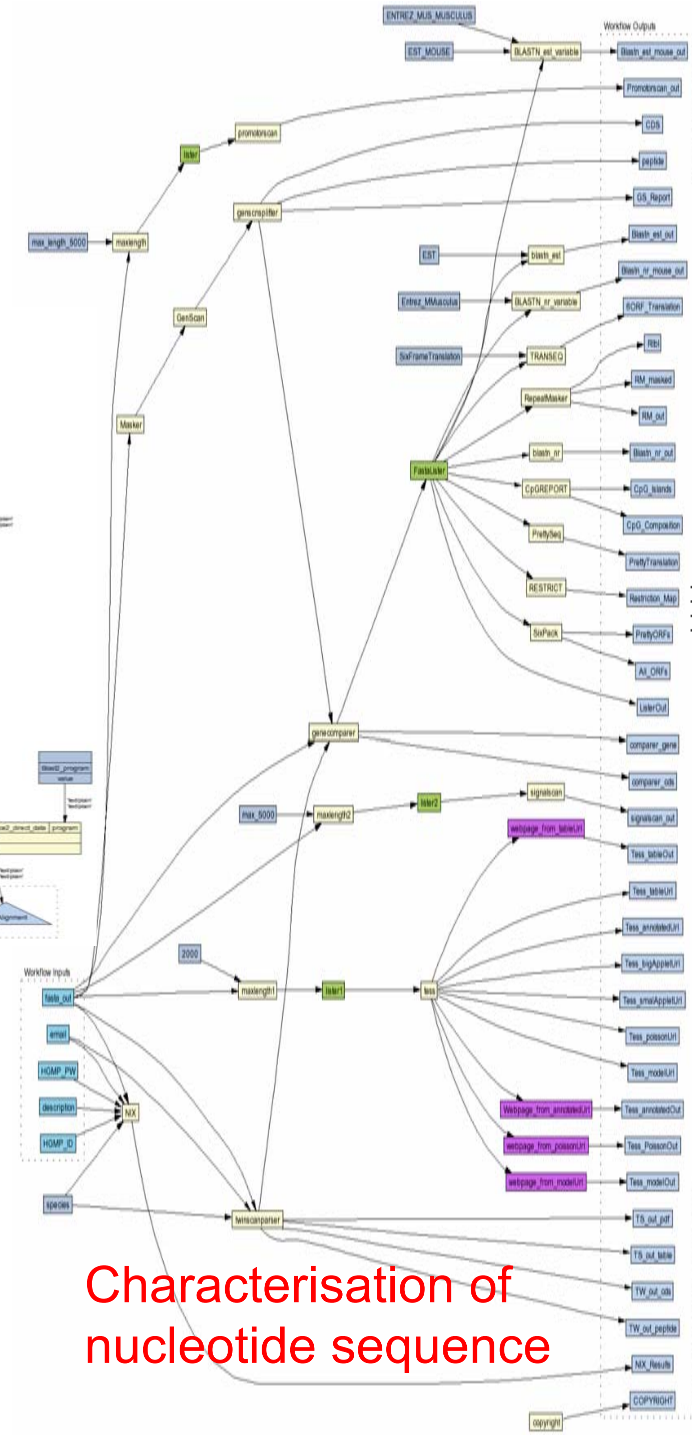


myGrid
Pink: Outputs/inputs of a service
Purple: Tailor-made services
Green: Emboss soaplab services
Yellow: Manchester soaplab services
Query nucleotide sequence
RepeatMasker
BLASTwrapper
GenBank Accession No
URL inc GB identifier
GenBank Entry
Sort for appropriate Sequences only
Seqret
Nucleotide seq (Fasta)
RepeatMasker
GenScan
Promotor Prediction
TF binding Prediction
Regulation Element Prediction
Identify regulatory elements in genomic sequence
Translation/sequence file. Good for records and publications
prettyseq
Amino Acid translation
Identifies PEST seq
epestfind
Identifies FingerPRINTS
pscan
MW, length, charge, pI, etc
pepstats
Predicts Coiled-coil regions
pepcoil
6 ORFs
tblastn Vs nr, est, est_mouse, est_human databases. Blastp Vs nr
BlastWrapper
Predicts cellular
SignalP TargetP PSORTII
Identifies functional and structural domains/motifs
InterPro
Hydrophobic regions
Pepwindow? Octanol?
Coding sequence
sixpack
transeq
ORFs
restrict
Restriction enzyme map
cpgreport
CpG Island locations and %
RepeatMasker
Repetitive elements
ncbiBlastWrapper
Blastn Vs nr, est databases.

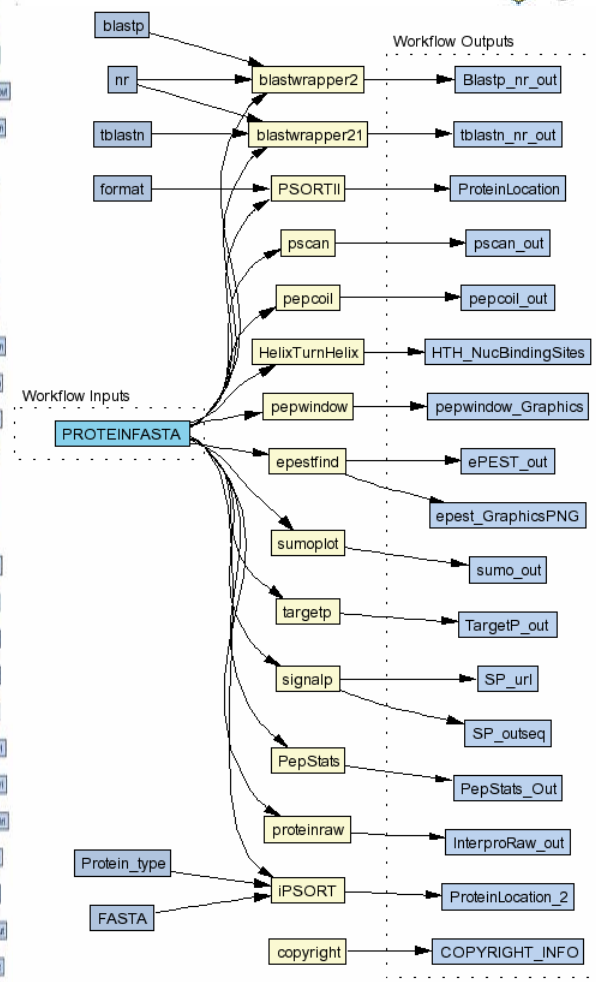
# Williams-Beuren Workflows



Identification of overlapping sequence

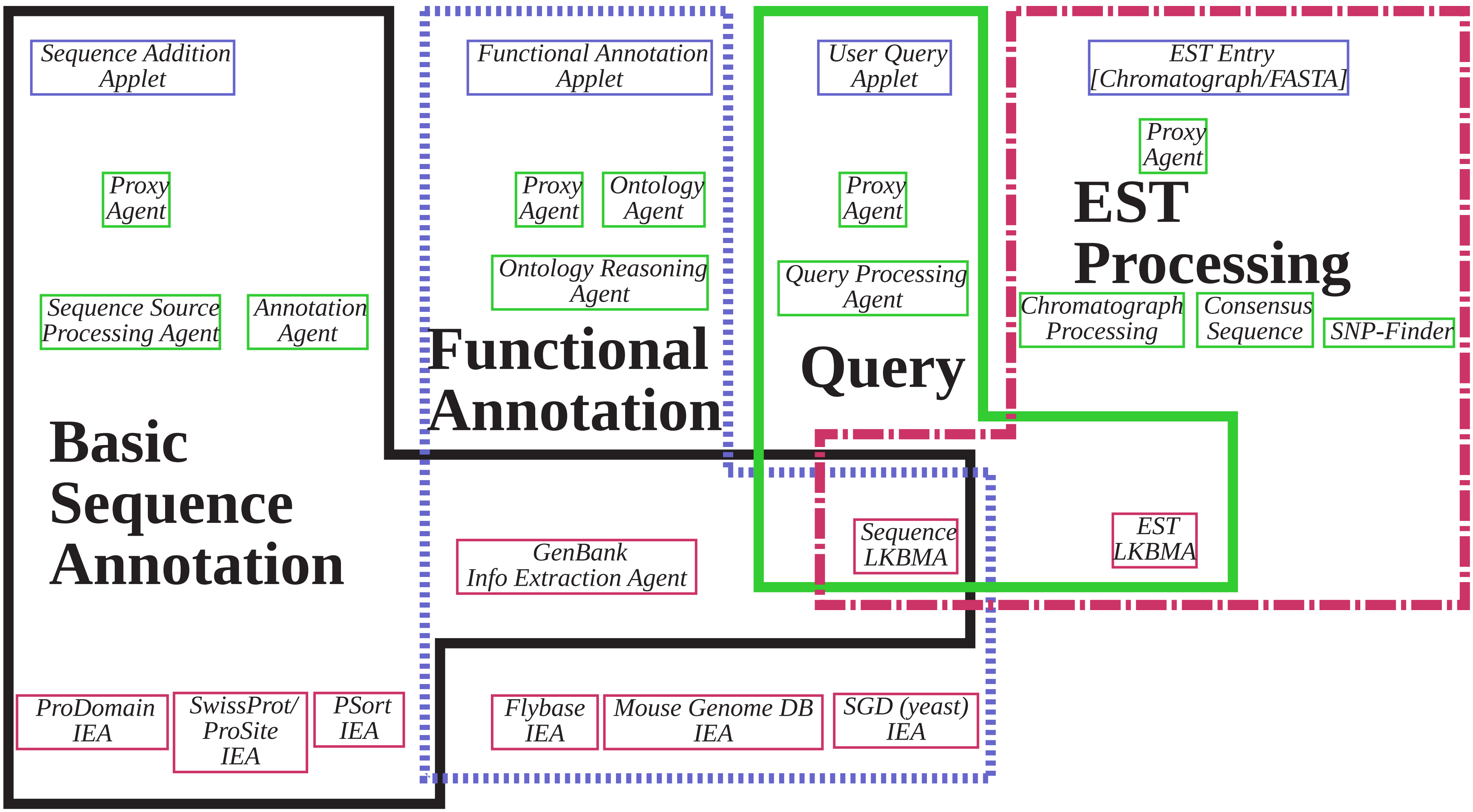


Characterisation of nucleotide sequence




Characterisation of protein sequence

# The BioMAS Genomic Annotation System

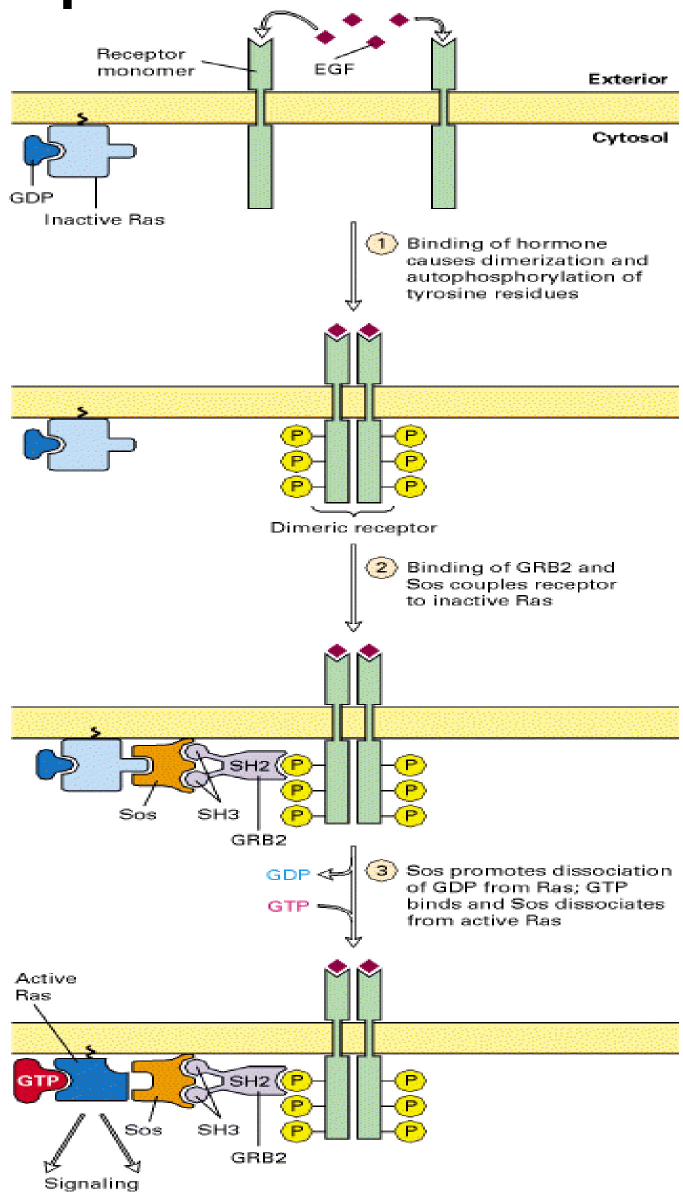


# Workflows and Multi-Agent Planning

- As of yet, very little automated creation of workflows (mostly built by hand)
- As of yet, very little automation of coordinated grid resource scheduling

# Biological Pathway Discovery thru AI Planning Techniques

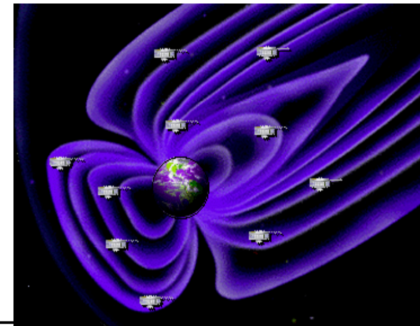
- To produce computer interpretable plans capturing relevant qualitative information regarding signal transduction pathways.
- To produce testable hypotheses regarding gaps in knowledge of the pathway, and drive future signal transduction research in an ordered manner.
- To identify key nodes where many pathways are regulated by a node with only 1 functional protein serving as a critical checkpoint.
- To perform in silico experiments of hyper expression and deletion mutation.
- To enable pathway vizualization tools by providing human- and machine-readable pathway description.

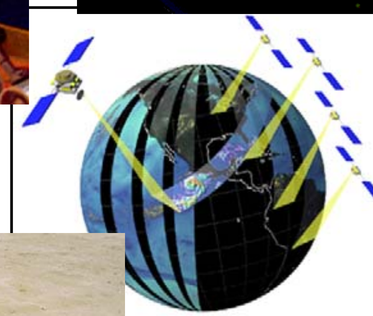
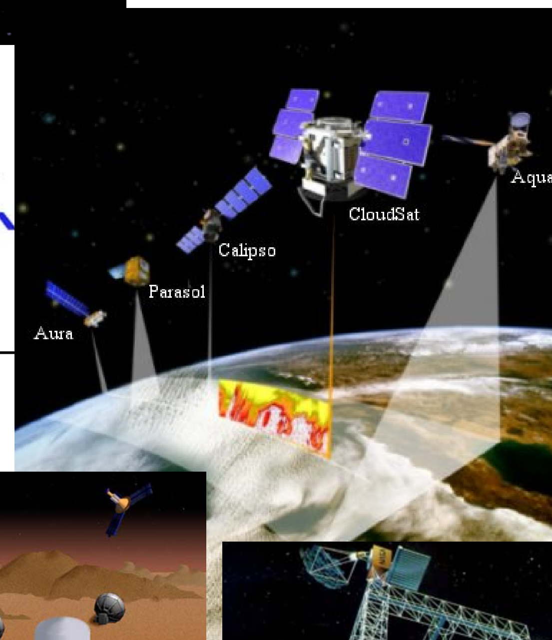


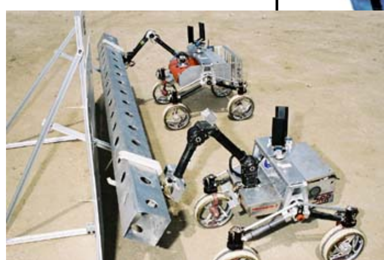
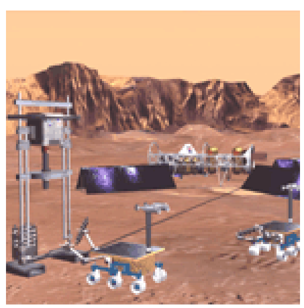
# Space Applications

**Decentralize decision-making?**

- **competing objectives (self-interest)**
- **control is already distributed**
- **communication constraints/costs**
- **computation constraints**

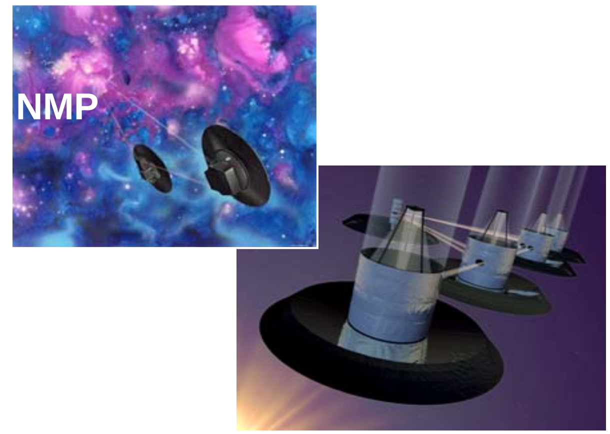
- multiple rovers
- spacecraft constellation
- Earth orbiters
- Mars network
- DSN antenna allocation
- mission planning
- construction, repair
- crew operations

# Applications – Multiple Spacecraft

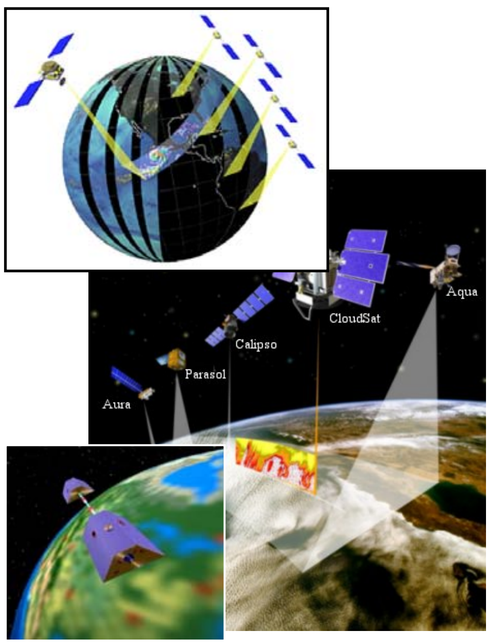
**Over 40 multi-spacecraft missions proposed!**

- Autonomous single spacecraft missions have not yet reached maturity.
- How can we cost-effectively manage multiple spacecraft?

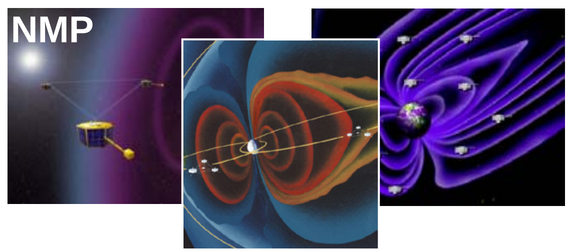
**Origins Program**



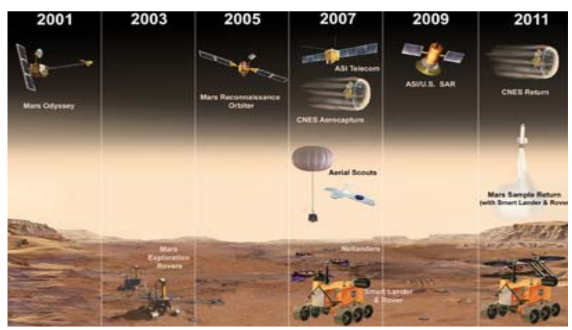

**Earth Observing System**



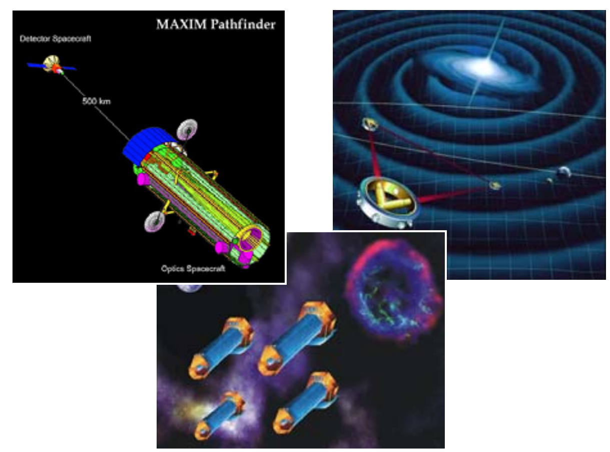

**Sun-Earth Connections**




**Mars Network**




**Structure & Evolution of the Universe**

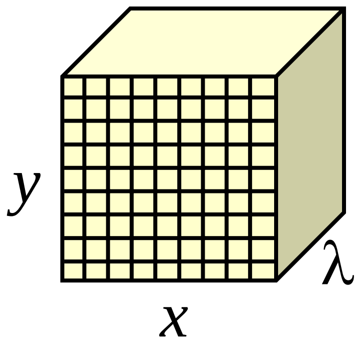


# Applications – Multiple Spacecraft

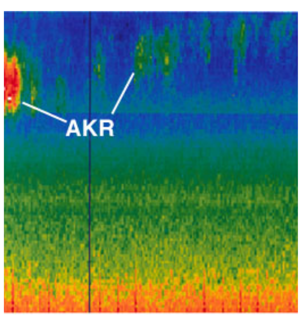
## Classification of Phenomena

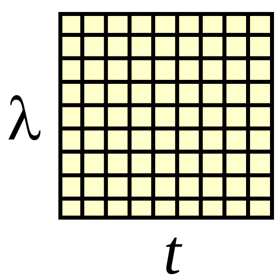
(Underlying Scientific Questions)

Signals from Celestial Sphere




Signals from Magnetosphere

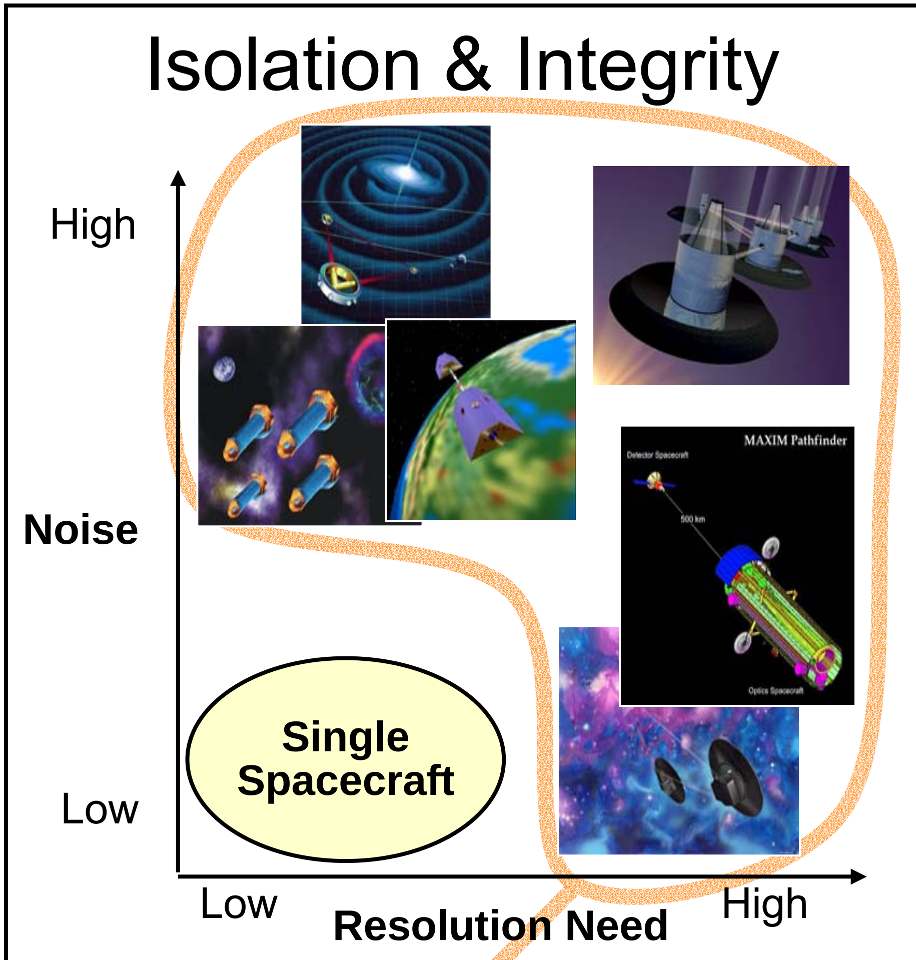


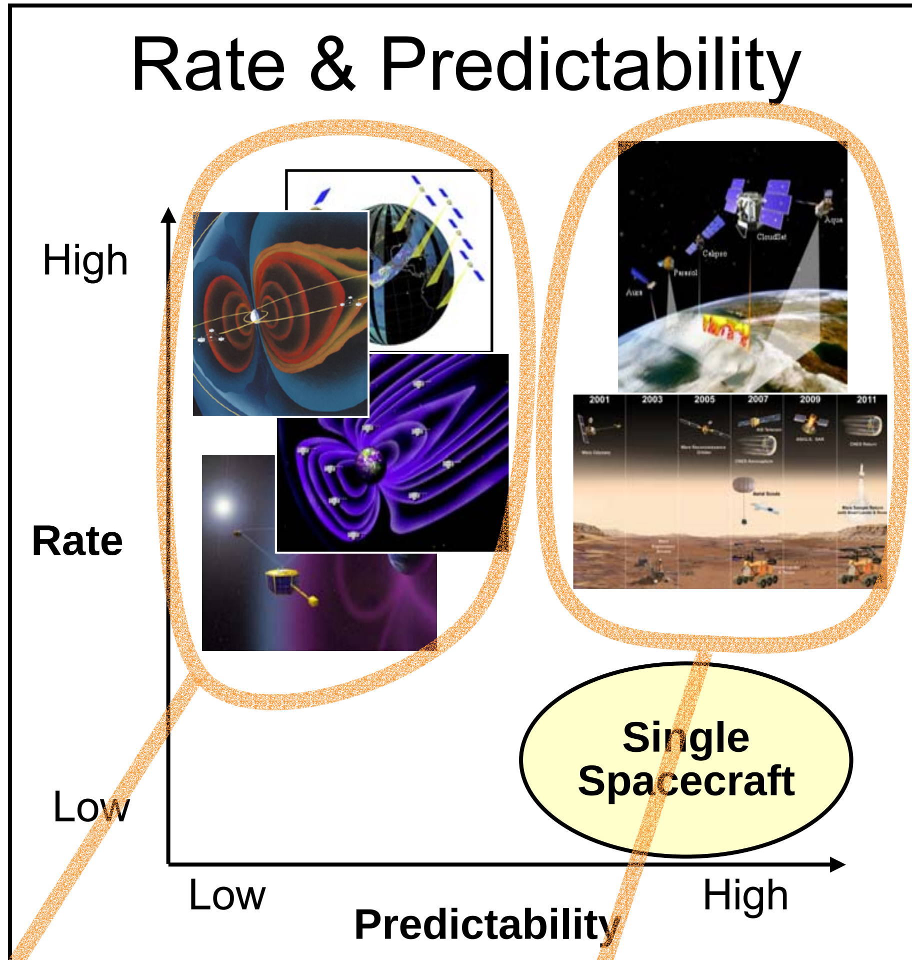




**Five Classification Metrics**

- Signal Location
  - **Where are the signals?**
- Signal Isolation
  - **How close are distinct signals in phenomenon?**
- Information Integrity
  - **How much noise is inherent in each signal?**
- Information Rate
  - **How fast do the signals change?**
- Information Predictability
  - **How predictable is the phenomenon?**

# Applications – Multiple Spacecraft Multiple Platform Mission Types



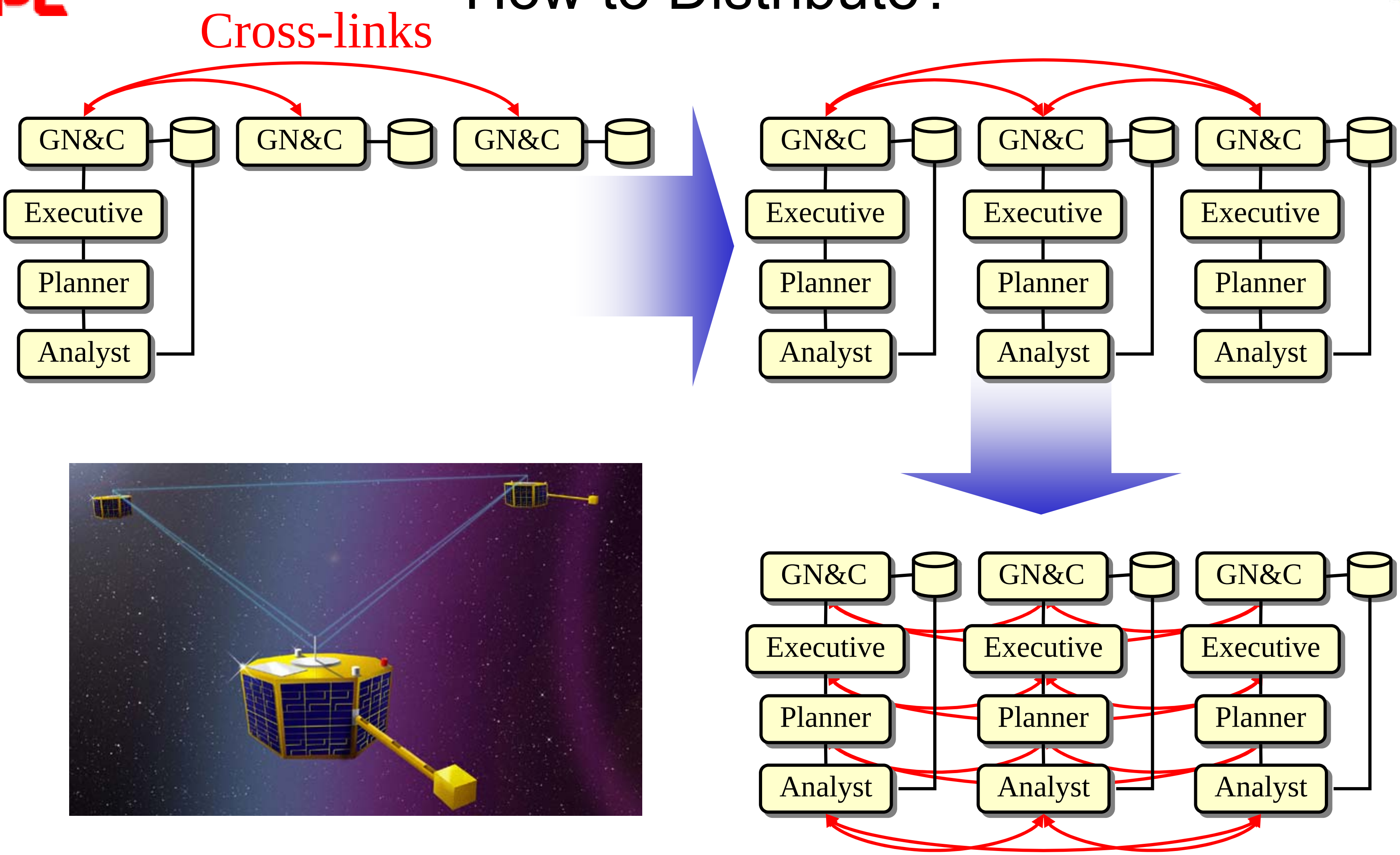

Signal Separation

Signal Space Coverage

Signal Combination

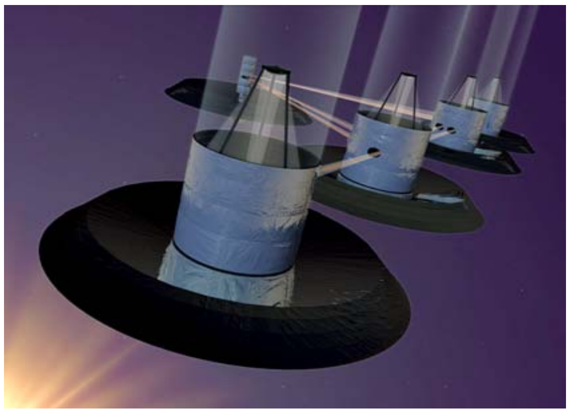
# Space Applications – Science

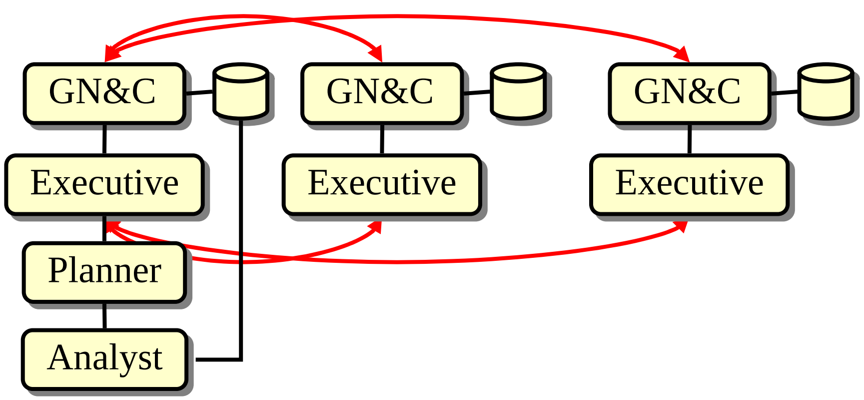
## How to Distribute?




*Who gets which components?*

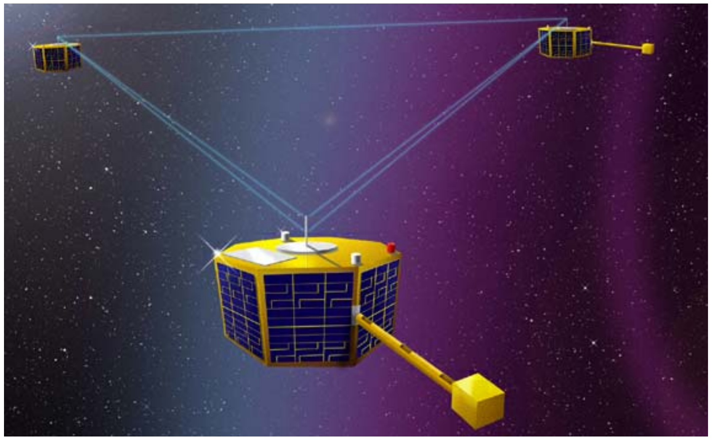
# Autonomous Signal Separation

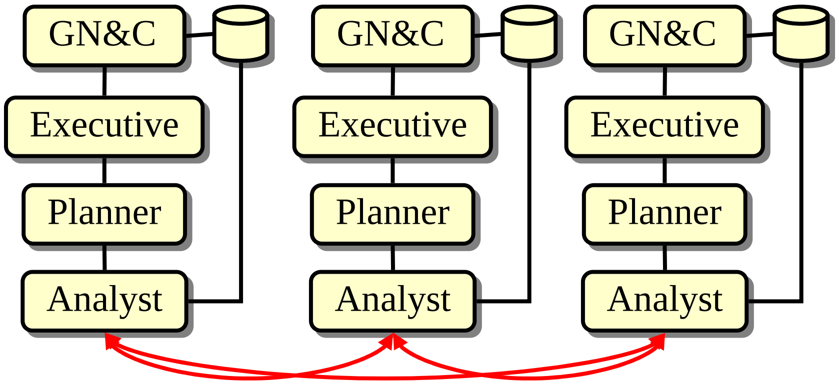





- Why many executives?
  - Each spacecraft can have local anomalies.
  - During an anomaly communications can be lost due to drift.
- Why only one planner?
  - During normal operations the spacecraft are guaranteed to be able to communicate.
  - Since spacecraft join to make an observation, only one analyst is needed.

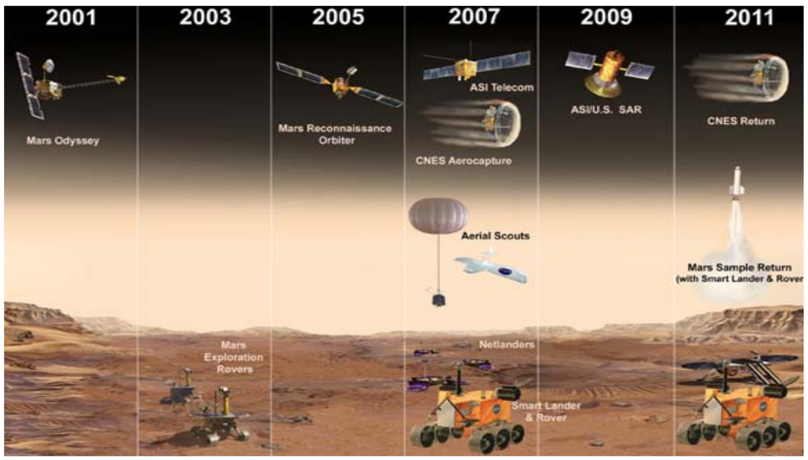
# Autonomous Signal Space Coverage

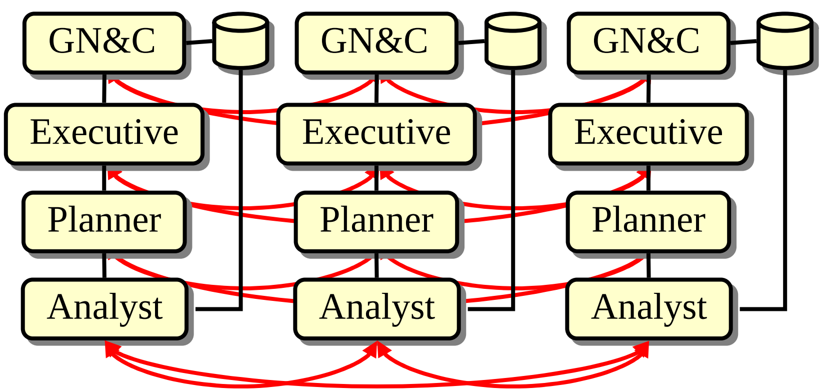





- Why many planners?
  - Cross-link is lost during normal operations, but spacecraft still have to manage local activities and respond to science events.
- Why communicate at all?
  - The value of local measurements is enhanced when combined with data from others.  Analysts must coordinate over collection.

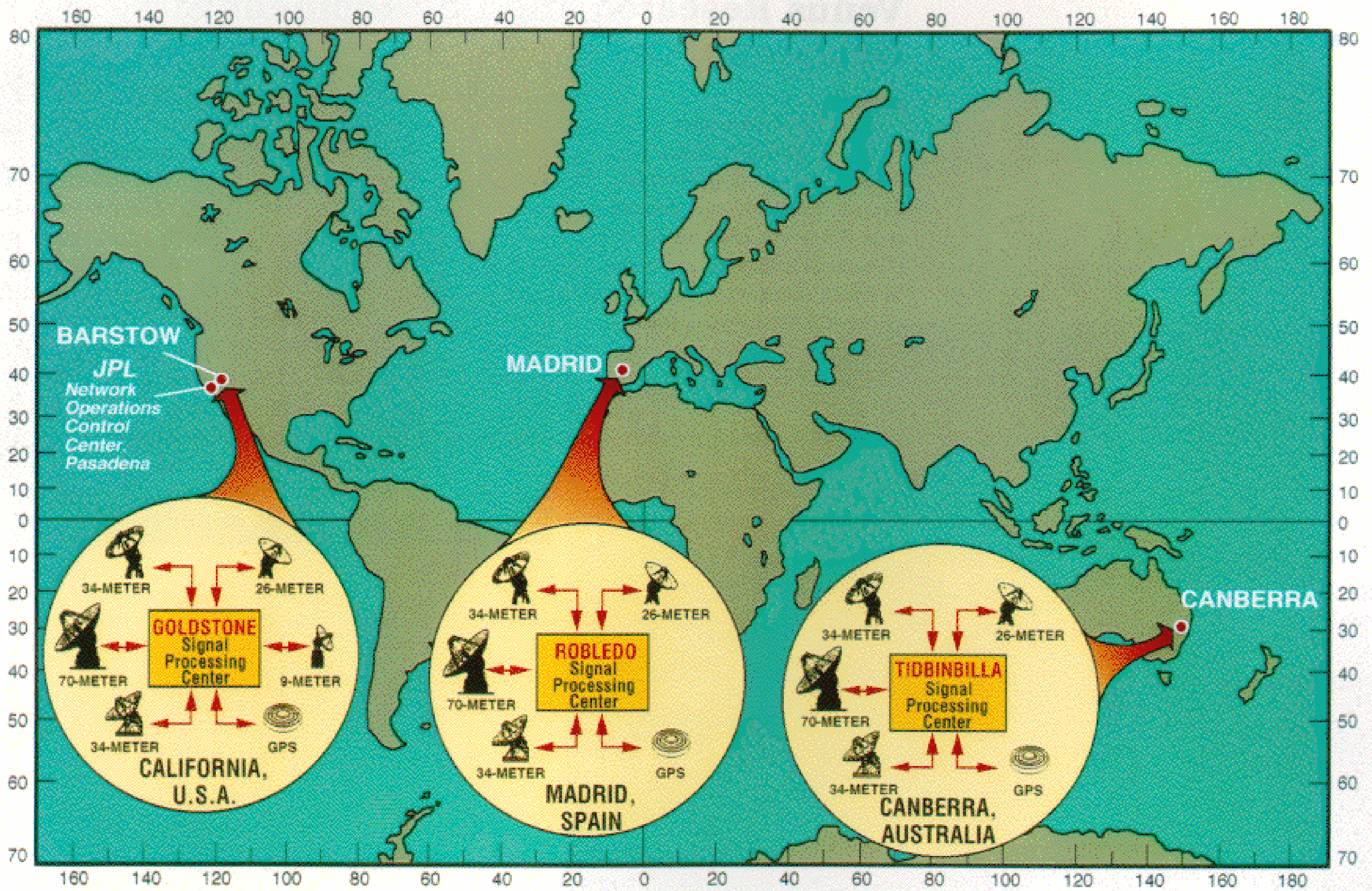
# Autonomous Signal/Mission Combination




- How does this differ from signal space coverage?
  - Each entity has different capabilities
    - Sensors: radar, optical, IR...
    - Mobility: satellite, rover...
    - Communications abilities.
  - Each mission has its own motivations.
    - There is a competition where each mission wants to optimize its own objectives in isolation.

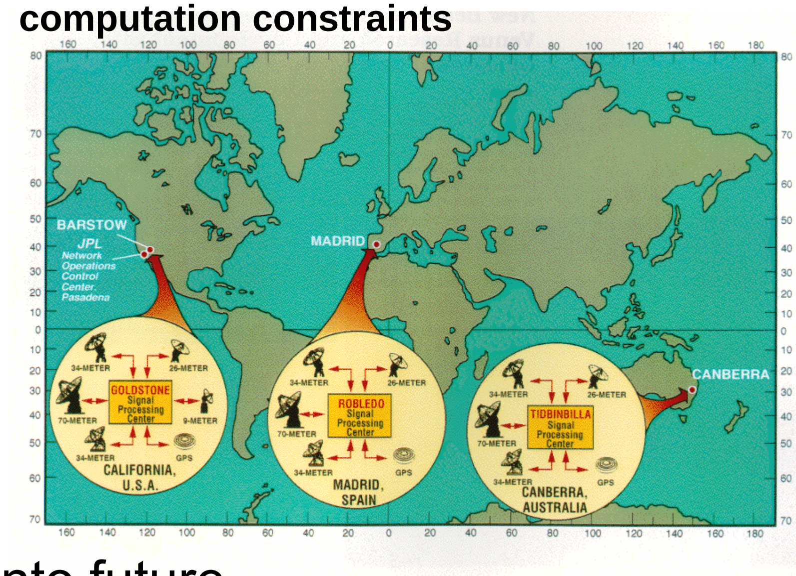
# Applications - Deep Space Network (DSN)



# Applications - Deep Space Network (DSN)

**Decentralize decision-making?**

- **competing objectives (self-interest)**
- **control is already distributed**
- **communication constraints/costs**
- **computation constraints**



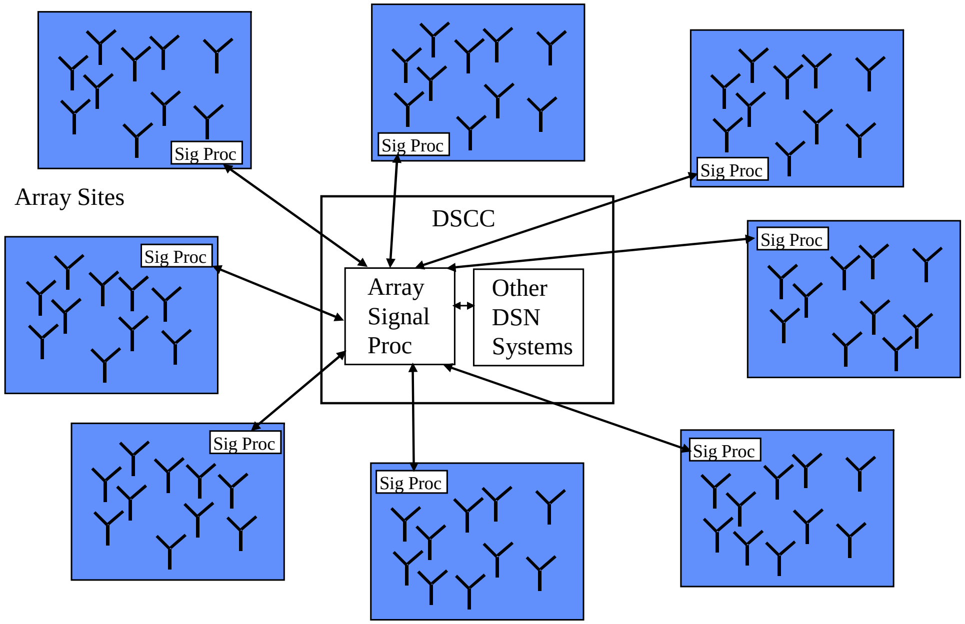

- 56 missions
- 12 antennas
  - different capabilities
  - shared equipment
  - geometric constraints
  - human operator constraints
- some schedule as long as 10 years into future
- some require schedule freeze 6 months out
- complicated requirements originally from agreement with NASA with flexibility in antennas, timing, numbers of *tracks*, gaps, etc.
- schedule centrally generated, meetings and horse trading to resolve conflicts
- similar to coordination operations across missions

# Applications – DSN Arrays

**Decentralize decision-making?**

- **competing objectives (self-interest)**
- **control is already distributed**
- **communication constraints/costs**
- **computation constraints**

- NASA may build 3600 10m weather-sensitive antennas
- 1200 at each complex in groups of 100 spread over wide area
- High automation requested—one operator for 100 or 1200 antennas
- Spacecraft may use any number of antennas for varying QoS, and may need link carried across complexes
- Only some subsets of antenna signals can be combined
  - depends on design of wiring/switching to combiners
  - combiners may be limited
- Local response time should be minimized

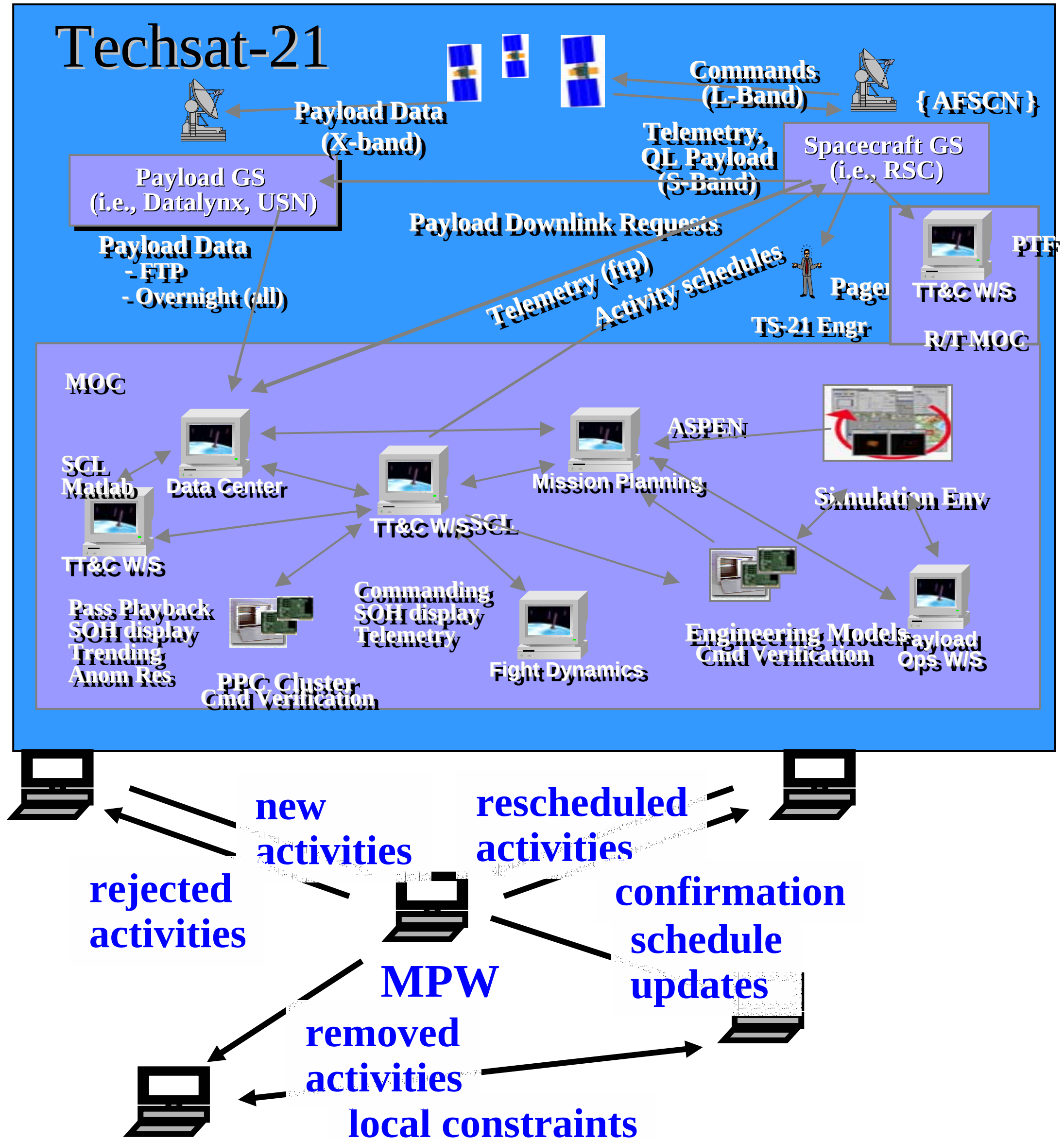


# Space Applications – Mission Operations

**Decentralize decision-making?**

- **competing objectives (self-interest)**
- **control is already distributed**
- **communication constraints/costs**
- **computation constraints**

- multiple instruments on spacecraft contend for resources
- multiple scientists may compete for one instrument (HST)
- scientists work with operations staff to make sure goals can be safely achieved
- plans must be validated (carefully simulated)
- changes made by users in parallel invalidate validation



# Game Applications

**Decentralize decision-making?**

- **competing objectives (self-interest)**
- **control is already distributed**
- **communication constraints/costs**
- **computation constraints**

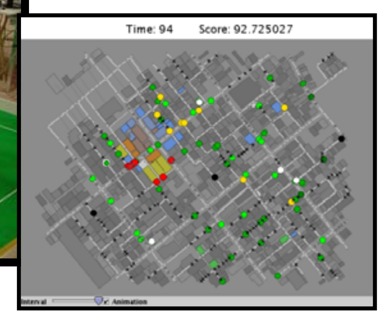
- FPS – first person shooter
  - e.g. Quake, Unreal
- RTS – real time strategy
  - e.g. Warcraft, Age of Empires, Freecraft)

# Game Applications

**Decentralize decision-making?**

- **competing objectives (self-interest)**
- **control is already distributed**
- **communication constraints/costs**
- **computation constraints**

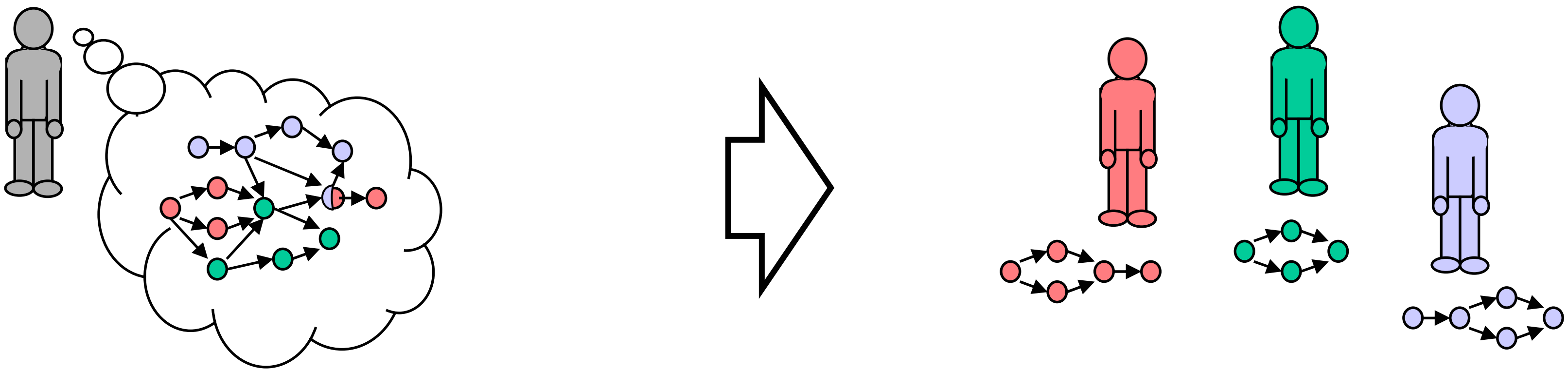

- MMORPG – massively multiplayer online role playing game
  - Ultima Online, Everquest, DAOC, …
- Robocup
  "By the year 2050, develop a team of fully autonomous humanoid robots that can win against the human world soccer champion team."

# Outline

- What is multi-agent planning?
- Design Issues
- Applications
- *Multi-agent planning problems and techniques*
  - Planning for multiple agents
  - Planning by multiple agents
  - Coordinating before planning
  - Coordinating plans of multiple agents
  - Planning and coordinating
  - Distributed continual planning

# Planning for Multiple Agents



- Centralized planning, decentralized execution
- Planning requires
  - concurrent activity
  - temporal expressivity
- Many planners can be used for this
  - SHOP, MIPS, TLPlan, LPG, ASPEN, Europa-2, *etc.*

# Centralized Planning for Distributed Plans

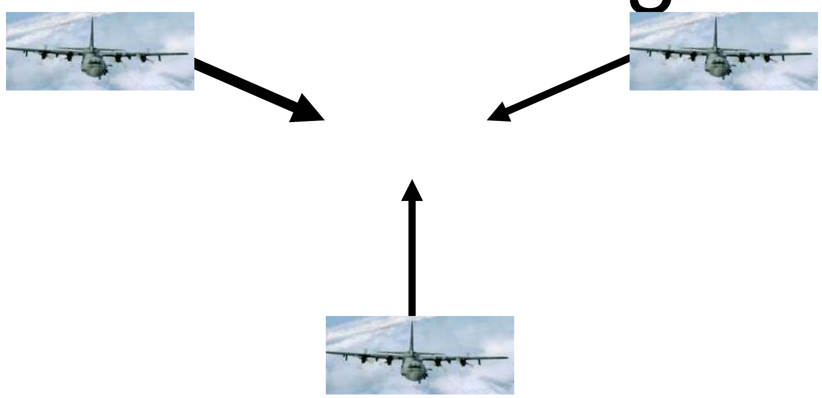
- Given a goal description, a set of operators, and an initial state description, generate a partial order plan
  - When possible, bias the search to find a plan in which the steps have few ordering constraints among them.
- Decompose the plan into subplans such that ordering relationships between steps tend to be concentrated within subplans and minimized across subplans.

# Centralized Planning for Distributed Plans

- Insert synchronization (typically, communication) actions into subplans.
- Allocate subplans to agents using task-passing mechanisms.
  - If failure, return to previous steps (decompose differently, or generate a different partial order plan,…).
  - If success, insert remaining bindings into subplans (such as binding names of agents to send synchronization messages to).
- Initiate plan execution, and optionally monitor progress
  - synthesize feedback from agents to ensure complete execution

# Task Centralization (Cammarata, McArthur, Steeb)

- Multi-agent planning:
  - construct a flight plan that will maintain an appropriate separation from other aircraft and satisfies other constraints (e.g. fuel consumption)
- Strategies for choosing centralized agent



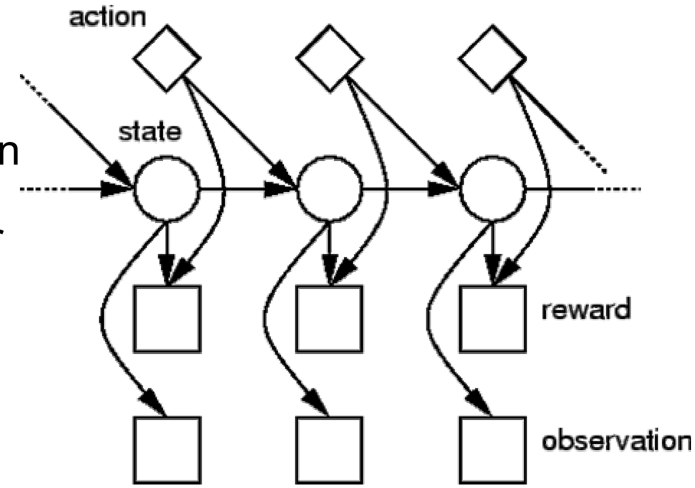
# Choosing Which Plane (Agent) to Resolve Conflict

- Selection by shared convention
  - no communication
- Selection of least spatially-constrained agent
  - agent which has most maneuverability
- Selection of most knowledgeable agent; least committed
  - agent which knows most about intentions of planes not directly involved in conflict

- Task Sharing
  - separate out decisions on who plans and who is planned

# Markov Decision Processes (MDPs)

- POMDPs – partially observable MDPs
  - *S* – states
  - *A* – actions, transition probabilities from $s_i$ to $s_j$ for $a_k$
  - *O* – observations, probabilities of obtaining observation $o_m$ when transitioning from $s_i$ to $s_j$ for action $a_k$
  - *V* – value function maps state history to a real number
- Extensions of MDPs for multiple agents
  - joint action
  - separate reward functions
  - observability by team
  - communication costs




| | Individually Observable | Collectively Observable | Collectively Partially Observable | Non-observable |
|---|---|---|---|---|
| No Comm. | MMDP | | Dec-POMDP POIPSG | |
| General Comm. | | Xuan-Lesser | COM-MTDP | |
| Free Comm. | MDP | MDP | POMDP | |

P-complete

NP-complete

PSPACE-complete

NEXP-complete

# References – MDPs for Agents

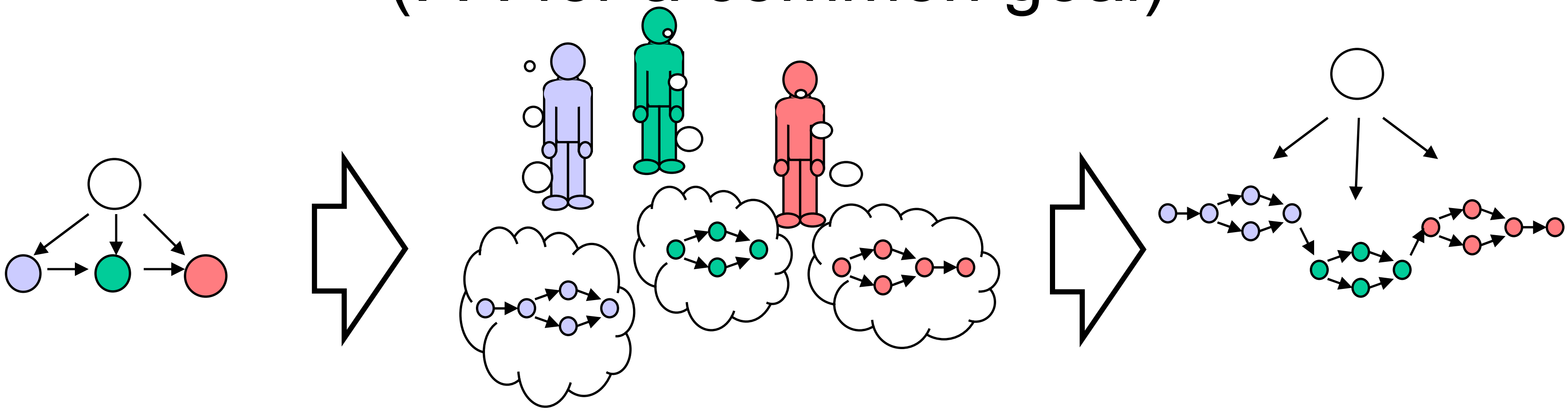
- MDPs – Boutilier, JAIR, 1999
- MMDP – Boutilier, IJCAI ’99
- Dec-POMDP – Bernstein *et al.*, UAI ’00
- Xuan & Lesser, Agents ’01, AAMAS ’02
- COM-MTDP, Pynadath & Tambe, AAMAS ‘02, JAIR ’02
- POIPSG, Peshkin *et al.*, UAI ‘01

# Stochastic Games (Self-Interested Multiagent POMDPs)

- Separate reward functions
- Policies can include games at each state
- Often applied to repeated matrix games
  - rock, paper, scissors (RPS)
- Mechanisms for coordination as plan unfolds?

Minimax-Q (Littman, 1994) and many others

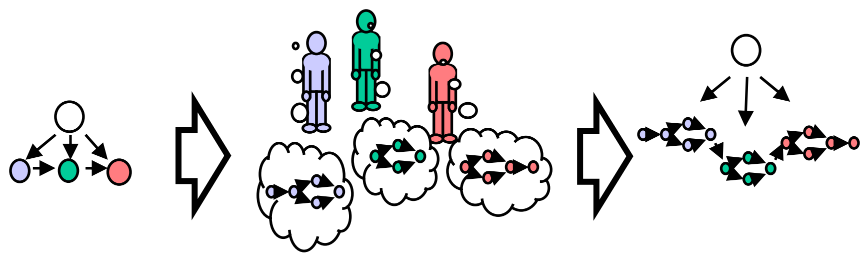
# Planning by Multiple Agents
## (. . . for a common goal)



- Cooperative
- Does not necessarily require
  - concurrent activity
  - temporal expressivity
- Overlaps with parallel algorithms/processing

# Distributed NOAH
# (Corkill, 1979)

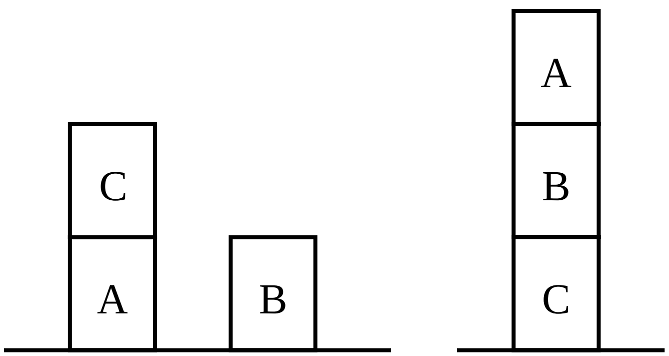
- Planning and execution by multiple agents



- Hierarchical planning
  - distribute conflict resolution (critic)
  - distribute world model
  - distribute resolution of deadlock
  - distribute elimination of redundant actions

# Distributed NOAH

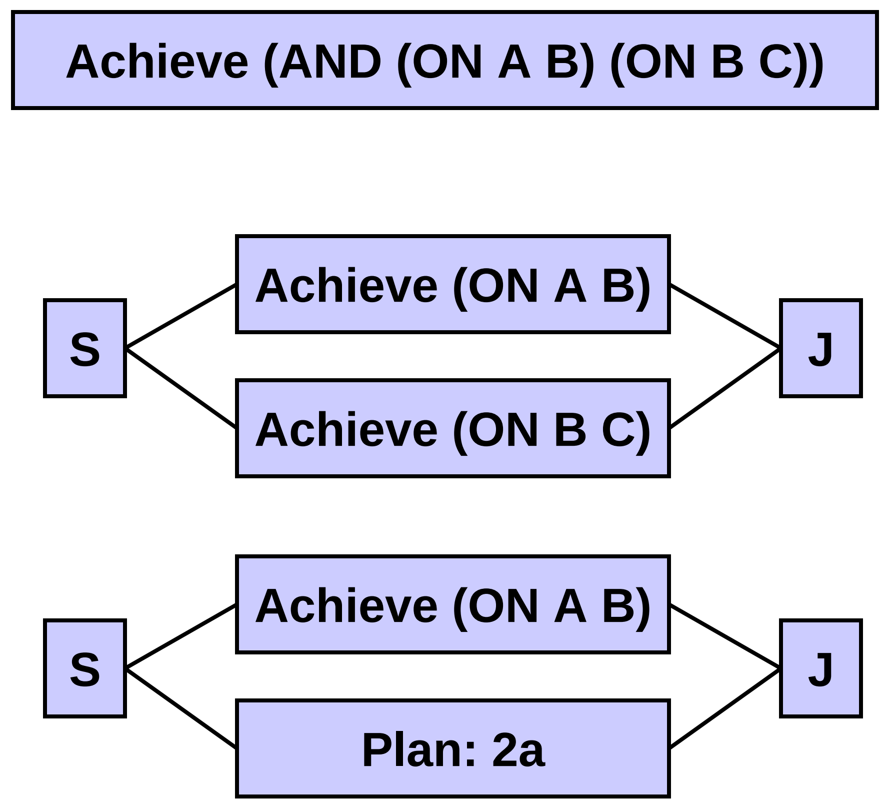
- Break conjunctive subgoals
- Transmit change in initial state as a result of local plans
- Recognize interactions and insert synchronization points



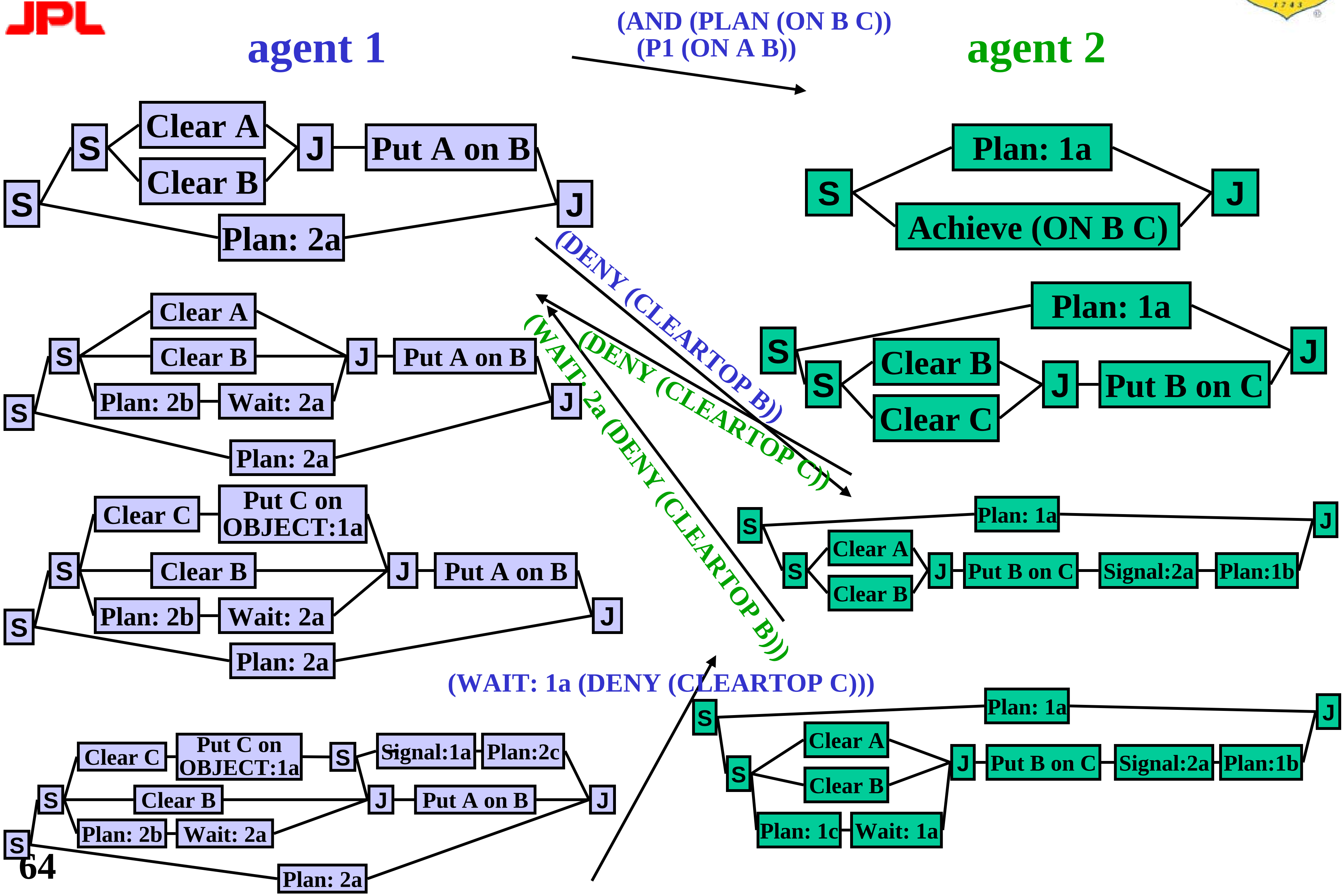

Initial State:
(ON C A)
(CLEARTOP B)
(CLEARTOP C)

Goal State:
(AND (ON A B)
ON B C))

agent 1



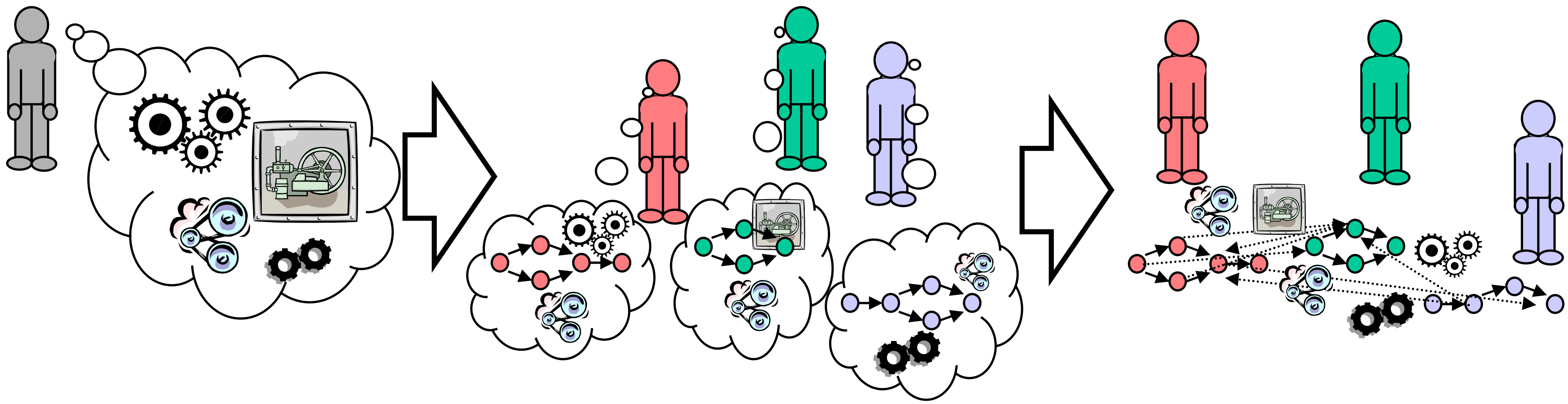
# Distributed NOAH

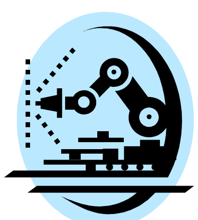


# COLLAGE
## (Lansky, 1991)

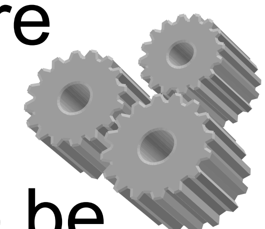
- Planning by multiple agents
- Distribute planning by partitioning into sub-problems
- Partially-ordered plan fragments with CSP-style binding constraints on action-parameter variables
- Action decomposition
- Planning as constraint satisfaction

# Coordinating Before Planning





- Centralized coordination, decentralized planning and execution
- Coordination is introduction (or creation) of mechanisms and/or constraints that ensure agents don't violate system constraints
- Could market mechanisms (e.g. auctions) be used this way?



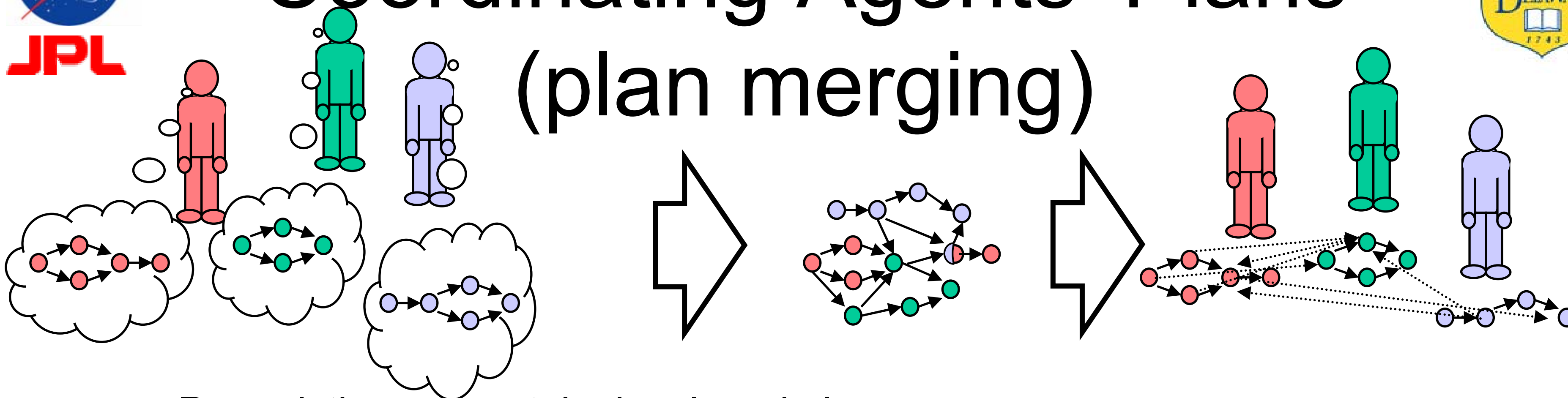
# Social Laws

(Shoham & Tennenholtz, 1992,
Briggs & Cook, 1995)

- constrain actions of agents with laws
- model of multi-agent action with social laws
- finding social laws is NP-Complete
- Example
  - traffic laws: space-time separation of mobile robots



# Coordinating Agents' Plans (plan merging)



- Pre-existing separately developed plans
- Goal is to resolve conflicts over states and resources and avoid redundant action
- Solutions are commitments in the form of
  - temporal constraints (requiring wait, signal actions)
  - subplan choices (e.g. drive or take taxi)
  - choices of effects on resources/states (e.g. use machine A instead of B)
- Assumes execution by agents, so need
  - concurrent action
  - temporal expressivity
- Can be centralized by communicating plans
- Much work
  - plan merging (Georgeff '83, Ephrati & Rosenschein '94, Tsamardinos, *et al*. '00)
  - hierarchical plan merging (Clement & Durfee, '99, Cox & Durfee, '03)

# Plan Merging

Given the candidate plans of the agents, consider all possible combinations of plans, executed in all possible orderings (interleavings or even simultaneous)

Generate all possible reachable sequences of states

For any illegal (inconsistent or otherwise failure) states, insert constraints on which actions are taken or when to ensure that the actual execution cannot fail

# Plan Merging Algorithm-1

**Each action has pre-conditions, post-conditions, and during-conditions (optional)**

- **Compare an agent's actions against each action of the other agents ($O(n^2a)$ comparisons) to detect contradictions between pre, post, and during conditions**
- **If none, pair of actions commute and can be carried out in any order.**
- **If some, determine if either can precede the other (post-conditions of one compatible with pre-conditions of other)**
- **All simultaneous or ordered executions not safe are deemed "unsafe"**

# Plan Merging Algorithm-2

**Ignore actions that commute with all others**

**Complete safety analysis by propagation**

- **Beginning actions a and b is unsafe if either consequent situation (adding post-conds of a to b, or b to a) leads to an unsafe ordering**
- **Beginning a and ending b is unsafe if ending a and ending b is unsafe**
- **Ending a and ending b is unsafe if both of the successor situations are unsafe**

# Plan Merging Algorithm-3

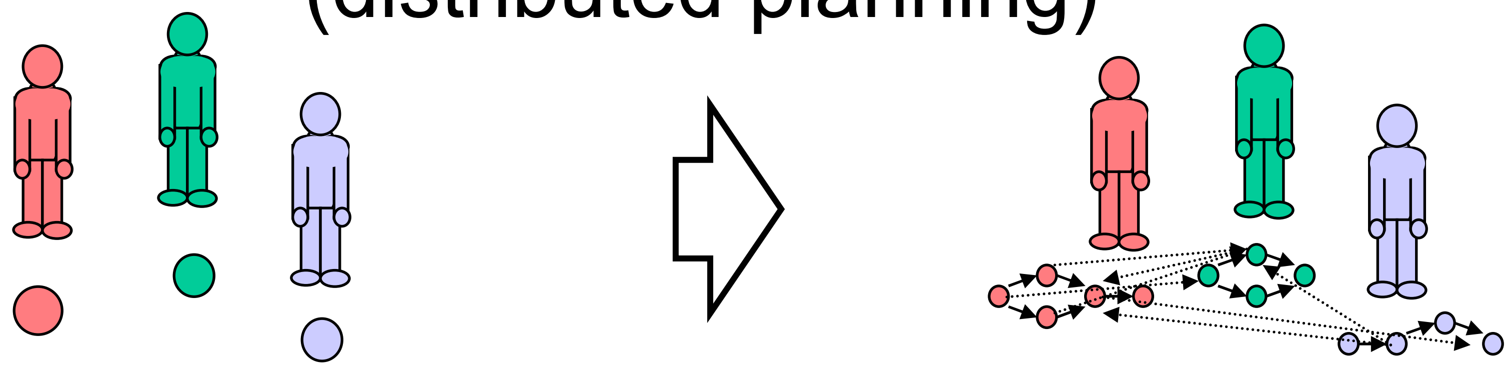
- In planning, assumption is that plan step interactions are exception
- Therefore, dropping commuting actions leaves very few remaining actions
- Examining possible orderings and inserting synchronization actions (messages or clock-times) therefore becomes tractable

# Iterative Plan Formation

**Sometimes, forming plans first and then coordinating them fails because of choices in initial plans formed**

**Instead, iterate between formation and coordination to keep alternatives alive**

# Planning and Coordinating
## (distributed planning)



- Same as prior case (coordinating agents' plans), but planning has not completed up front
- Opportunity to resolve conflicts as plans are being refined
- Should compare to prior case where plans developed without communication and then coordinated
- Decentralized decision-making
  - communication costs can dominate

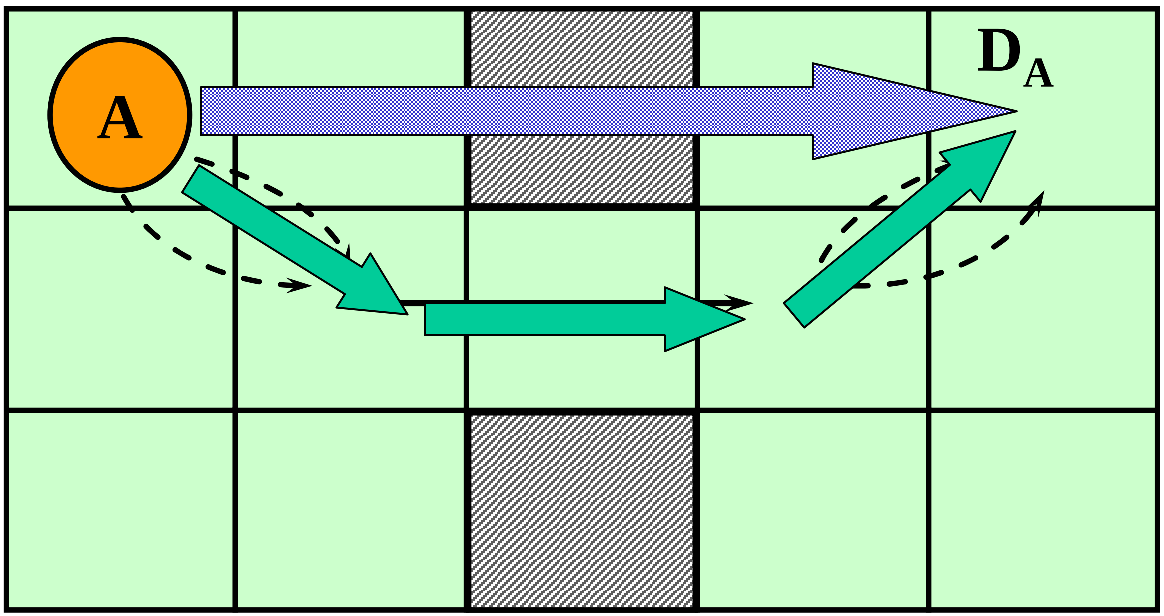
# Plan Combination Search

**Given initial propositions about the world**

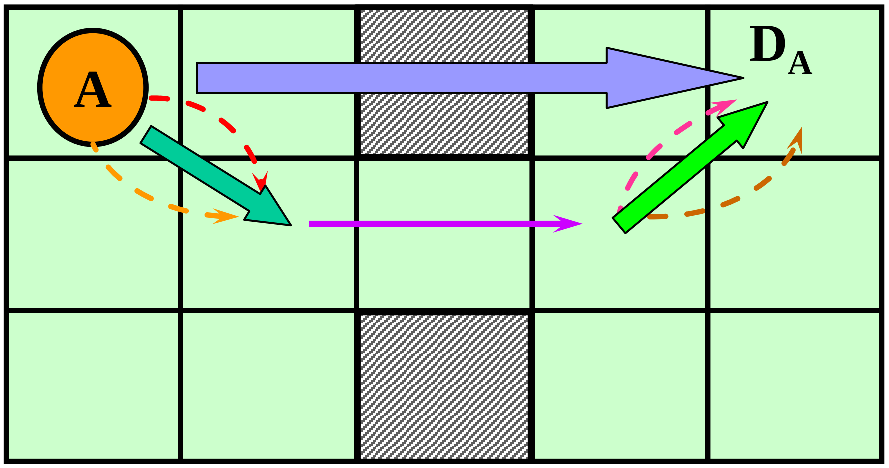
1. **Agents form successor states by proposing changes to current propositions caused by one action (or no-op)**
2. **Successor states are ranked using A* heuristic by all agents, and best choice is found and further expanded**

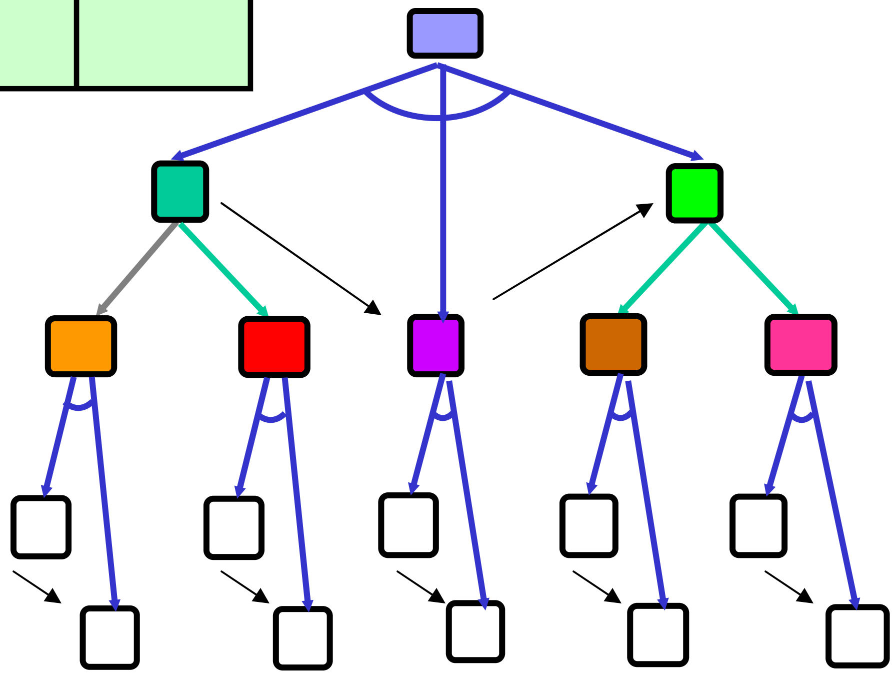
**Agents are simultaneously committing to a plan (corresponding to actions in solution path) and synchronizations (when actions are taken relative to each other)**

# Hierarchical Example

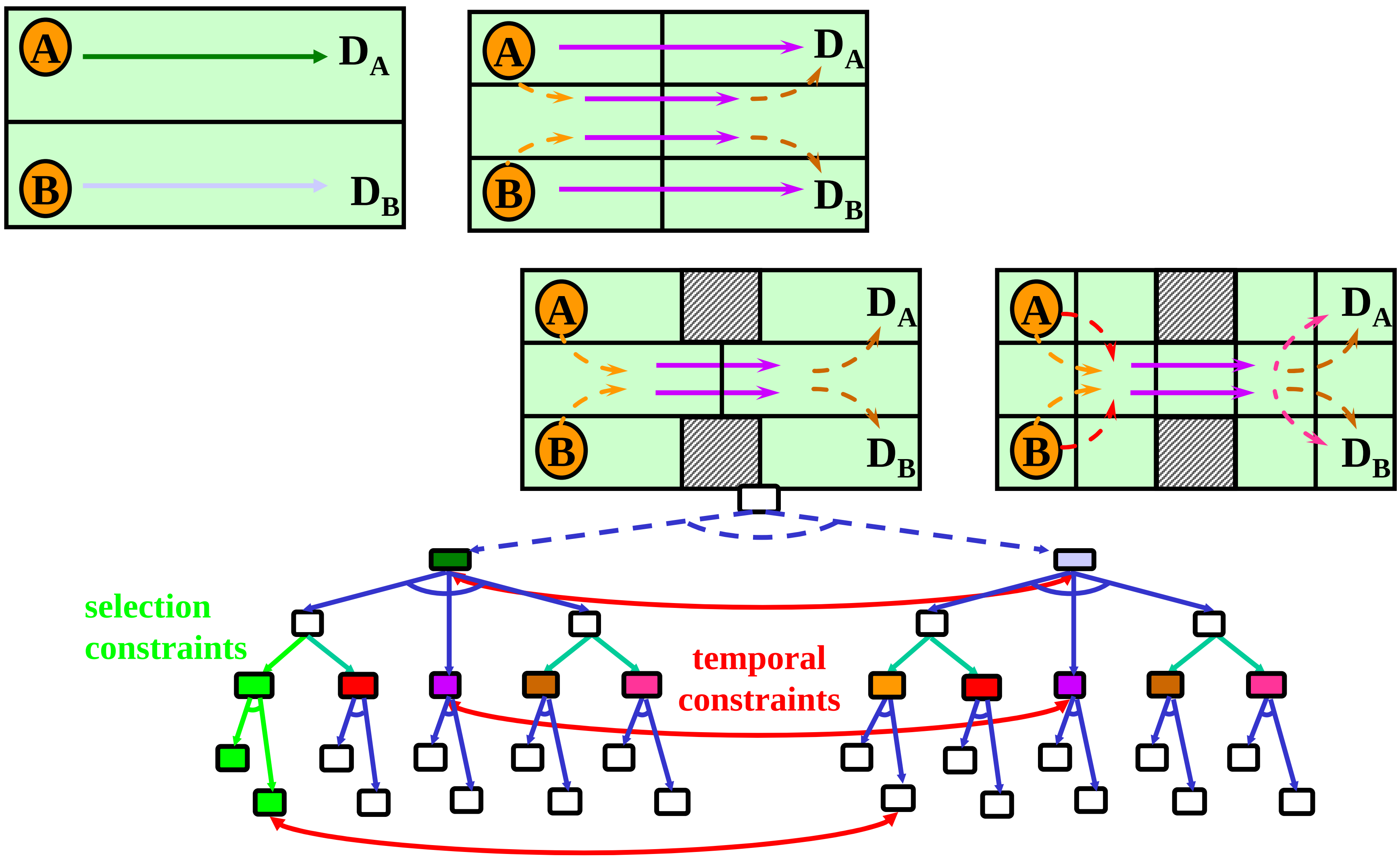


# Hierarchical Plan



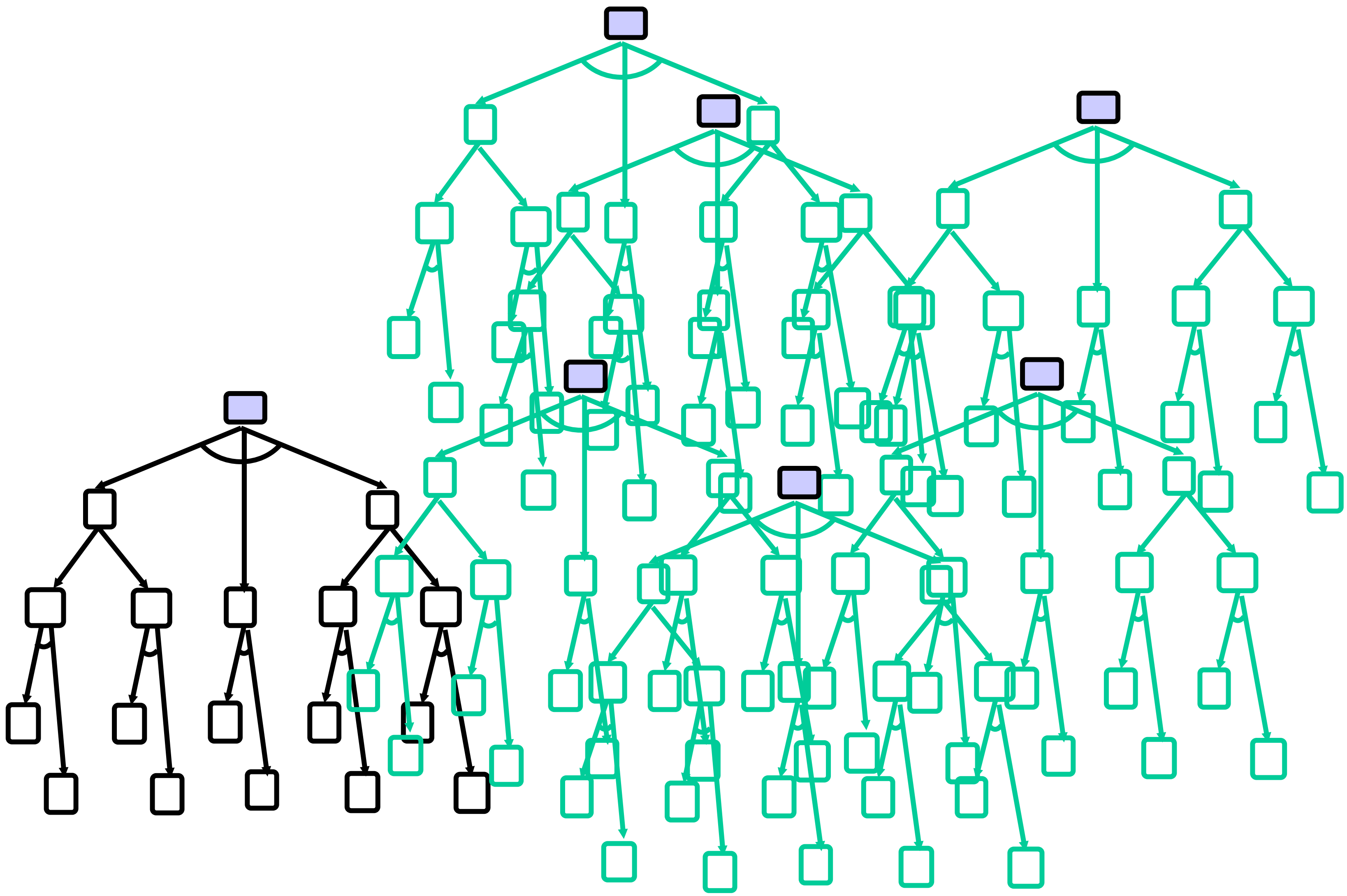
# Multi-level Coordination & Planning (Clement & Durfee, 1999)



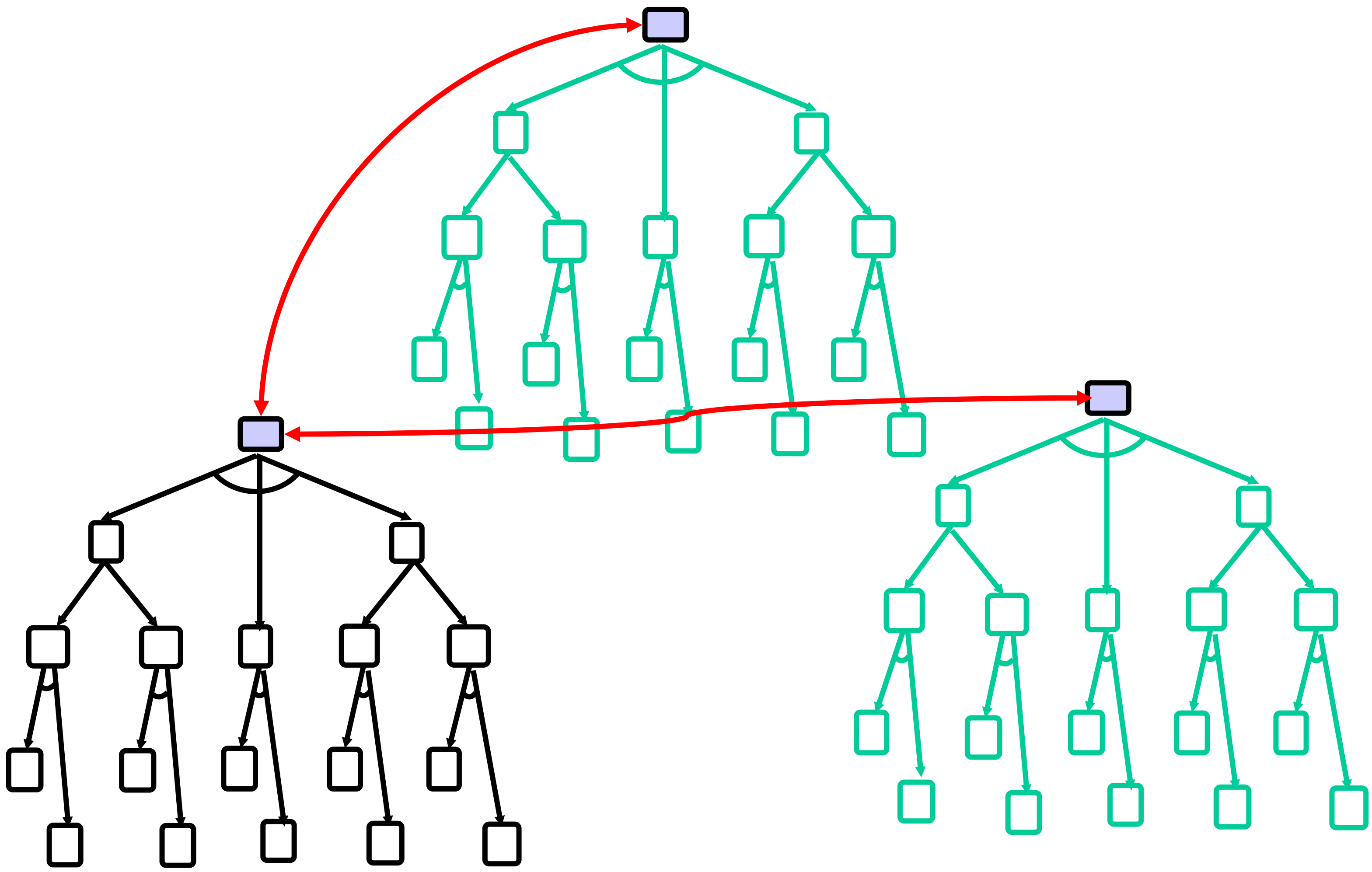
# Hierarchical Coordination Search

1. **Initialize the current abstraction level to most abstract**
2. **Agents exchange descriptions of their plans and goals at the current level**
3. **Remove plans or plan steps with no potential conflicts. If nothing left, done. If conflicts should be resolved at this level, skip next step.**
4. **Set the current level to the next deeper level, and refine all remaining plans (steps). Goto 2.**
5. **Resolve by: (i) put agents in a total order; (ii) current top agent sends its plans to others; (iii) lower agents change plans to avoid conflicts with received plans; (iv) next lower agent becomes top agent**

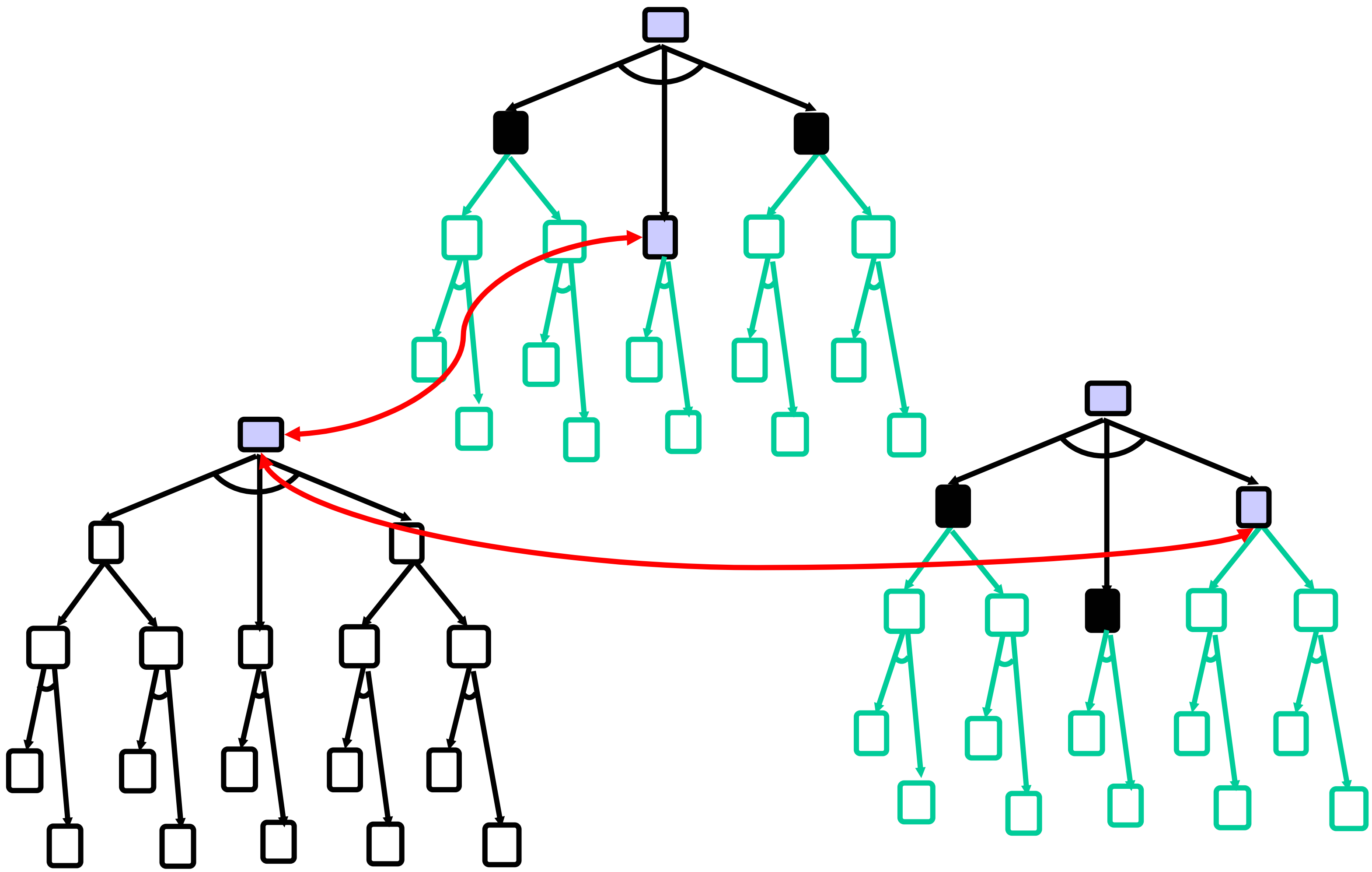
# Top-Down Coordination



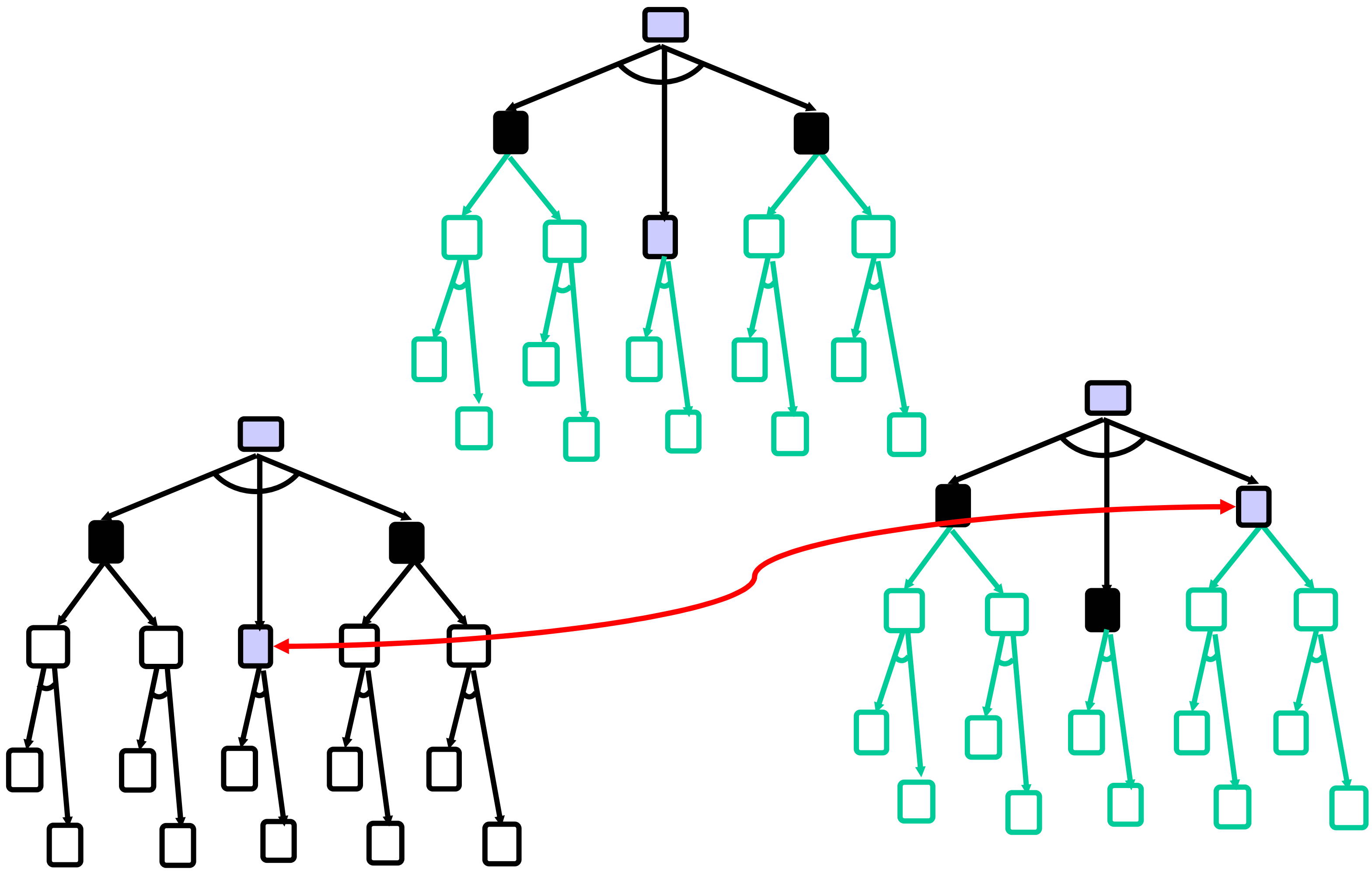
# Top-Down Coordination



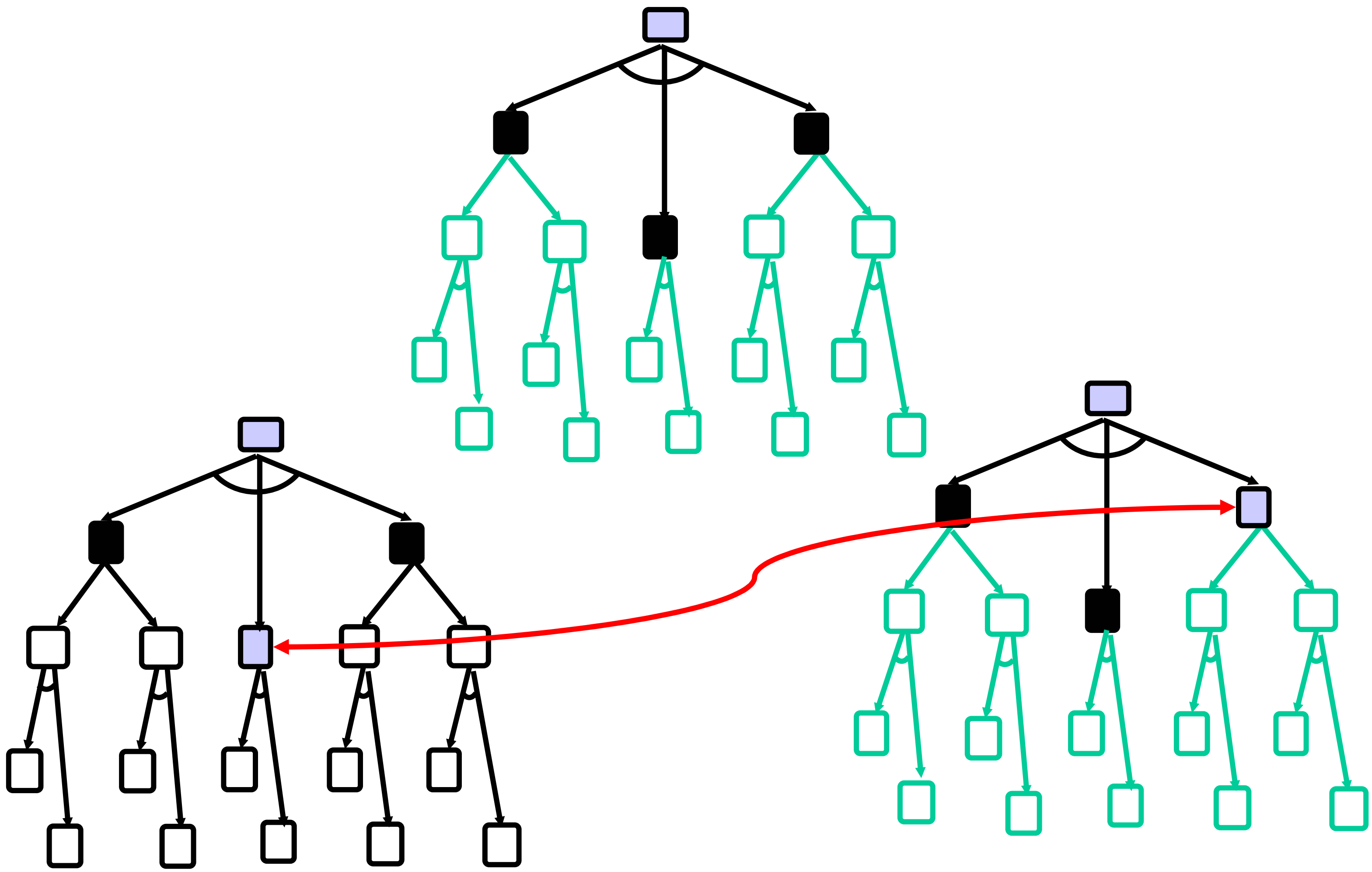
# Top-Down Coordination



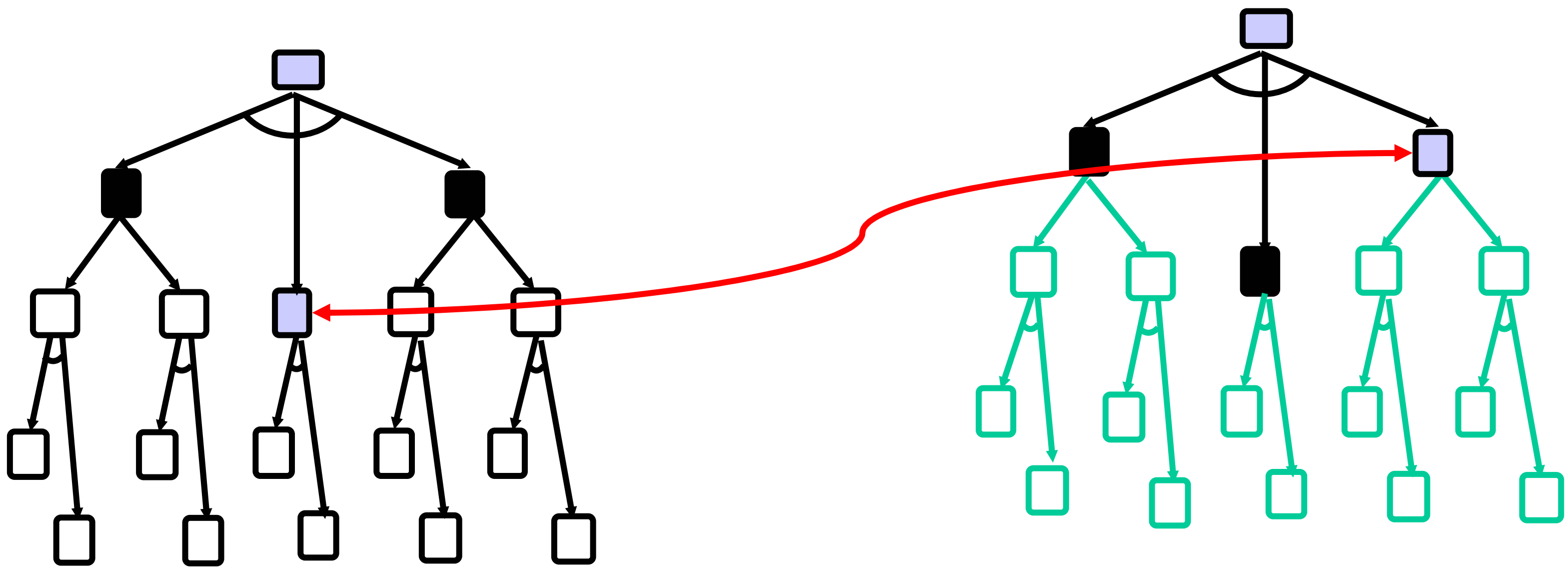
# Top-Down Coordination



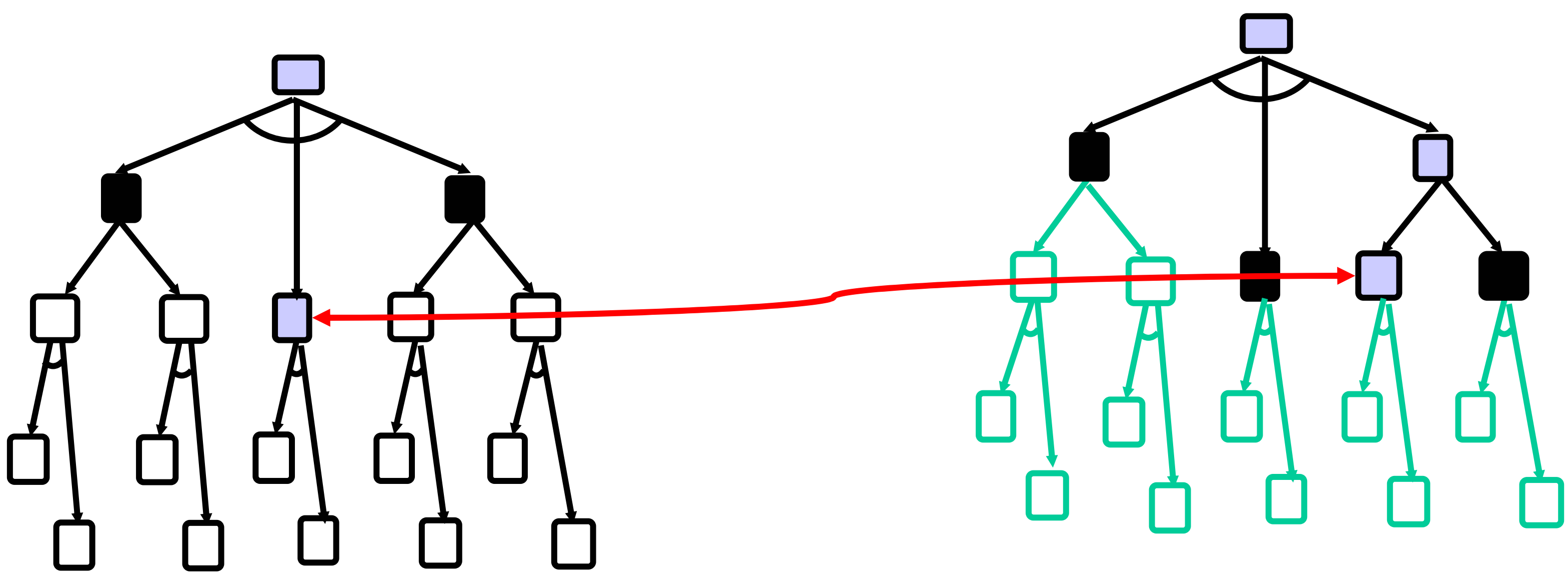
# Top-Down Coordination



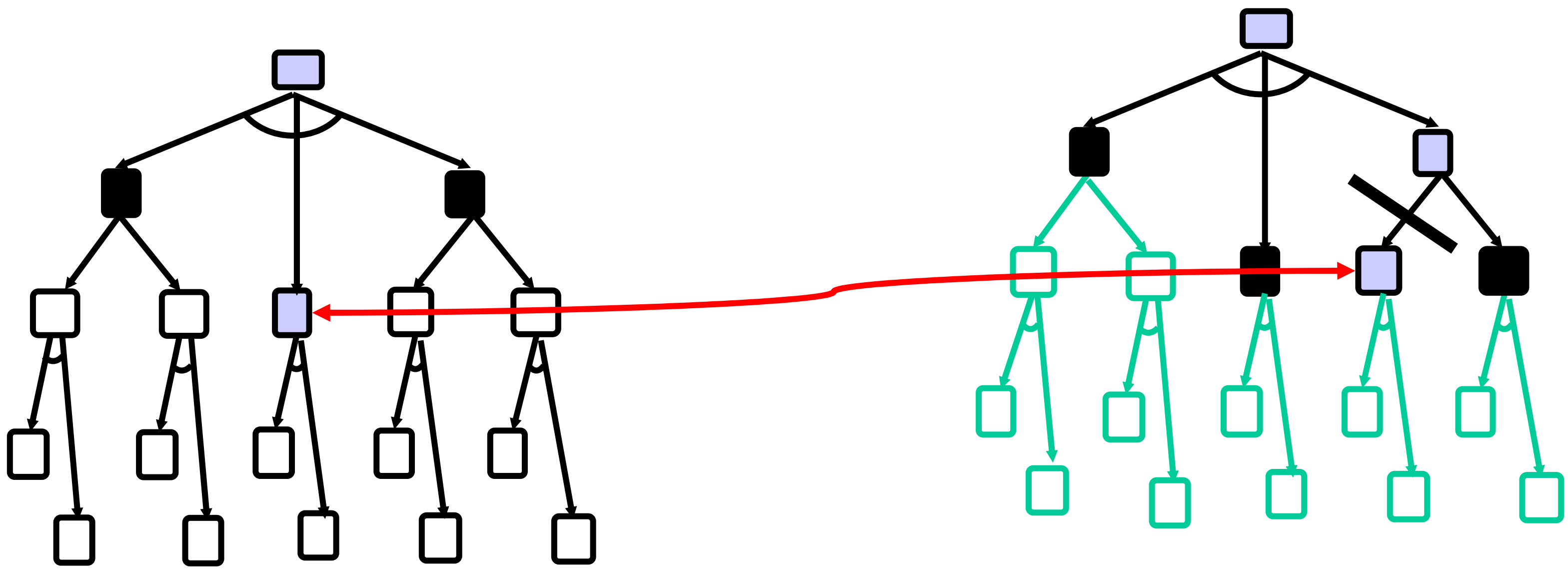
# Top-Down Coordination



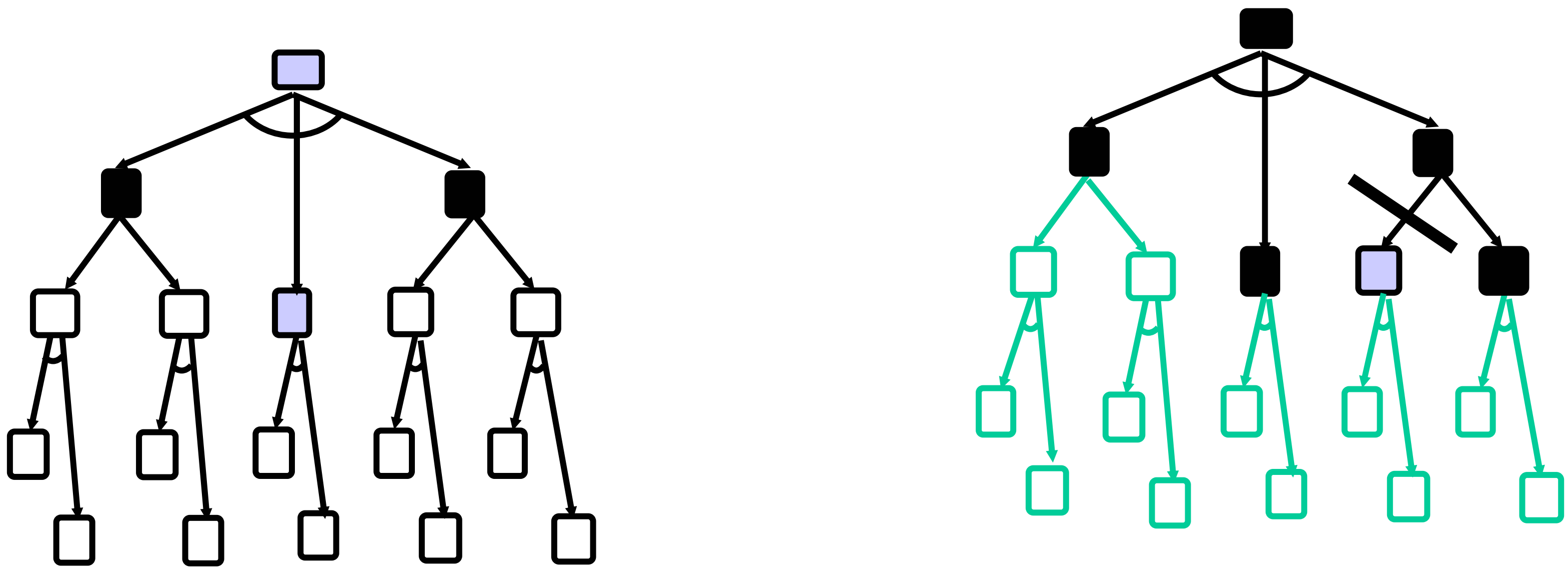
# Top-Down Coordination



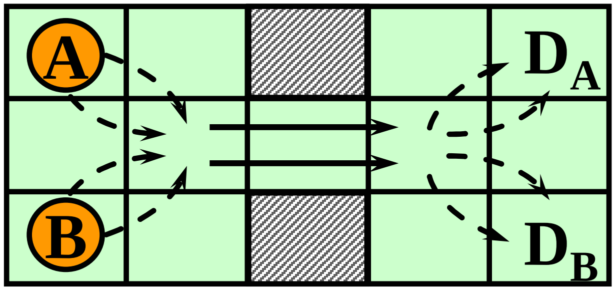
# Top-Down Coordination



# Top-Down Coordination

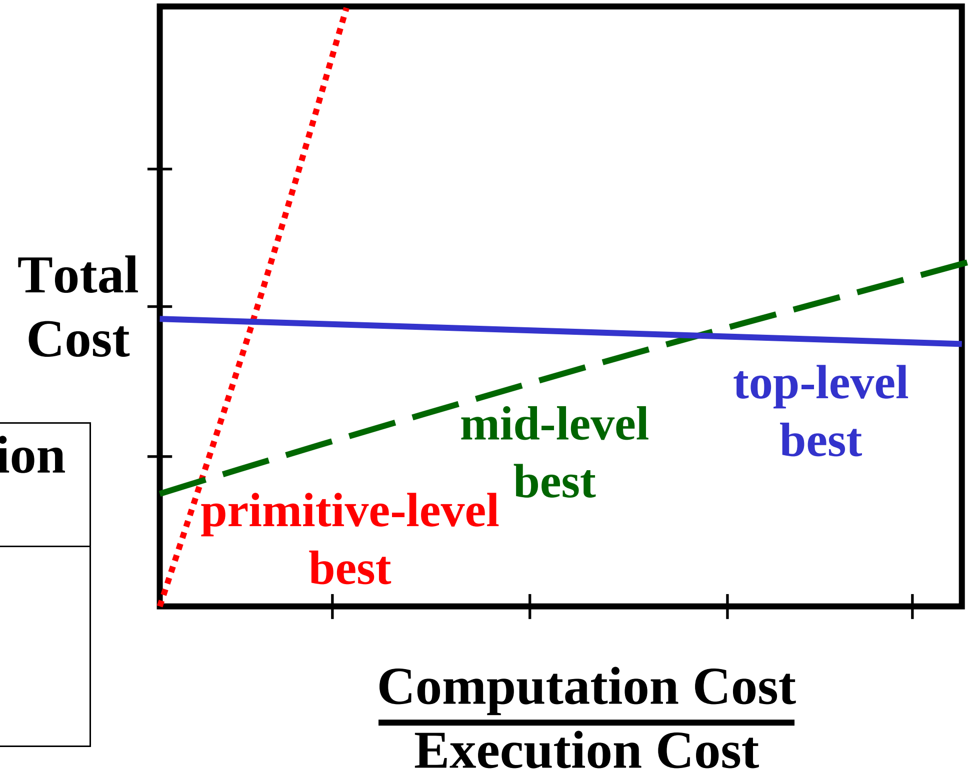


# Coordinating at Abstract Levels Can Improve Performance




## BFS algorithm

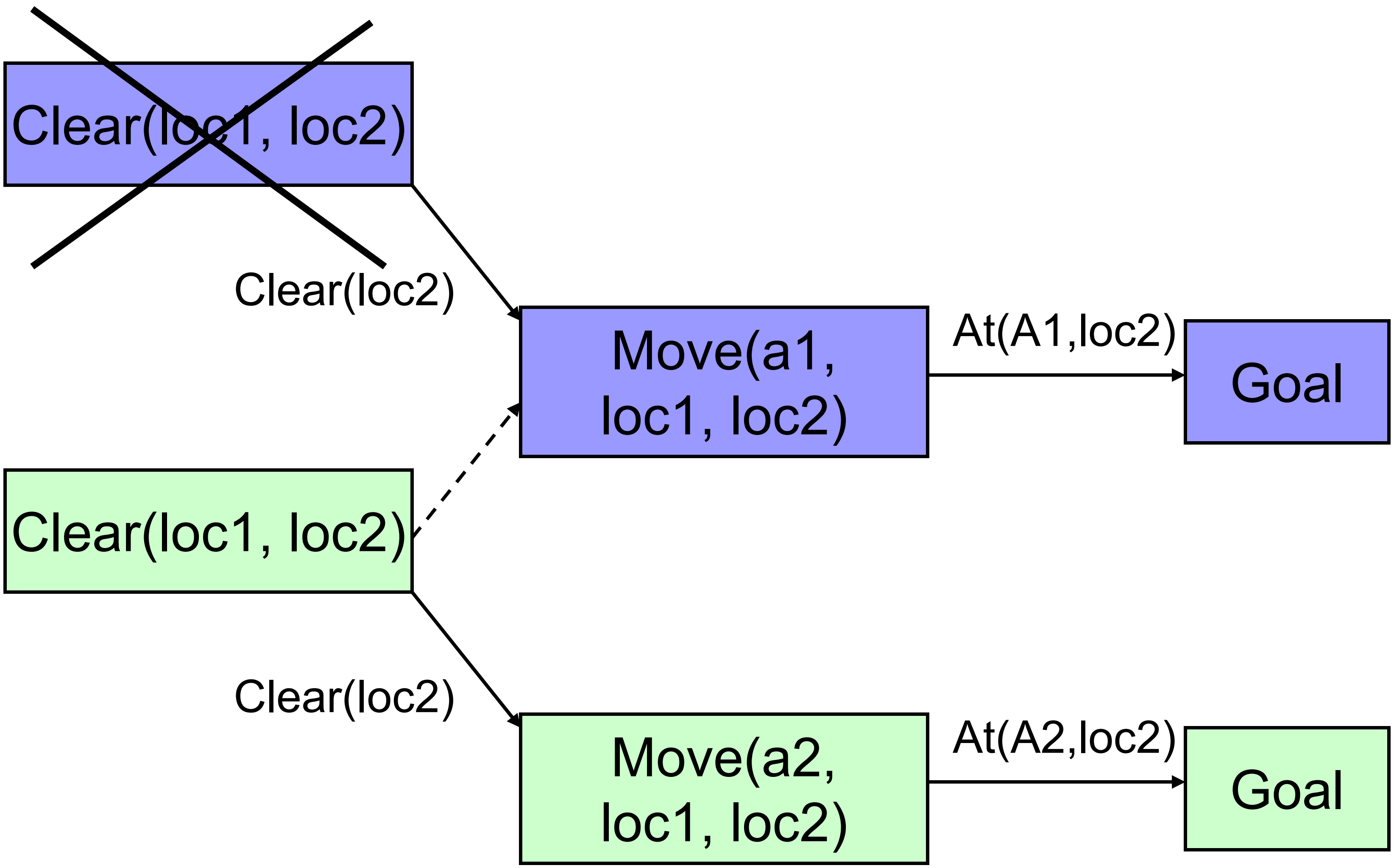
| level | computation time | execution time |
|---|---|---|
| top | 4 | 60 |
| mid | 159 | 40 |
| primitive | 2375 | 35 |




Exponential speedups unless three conditions hold

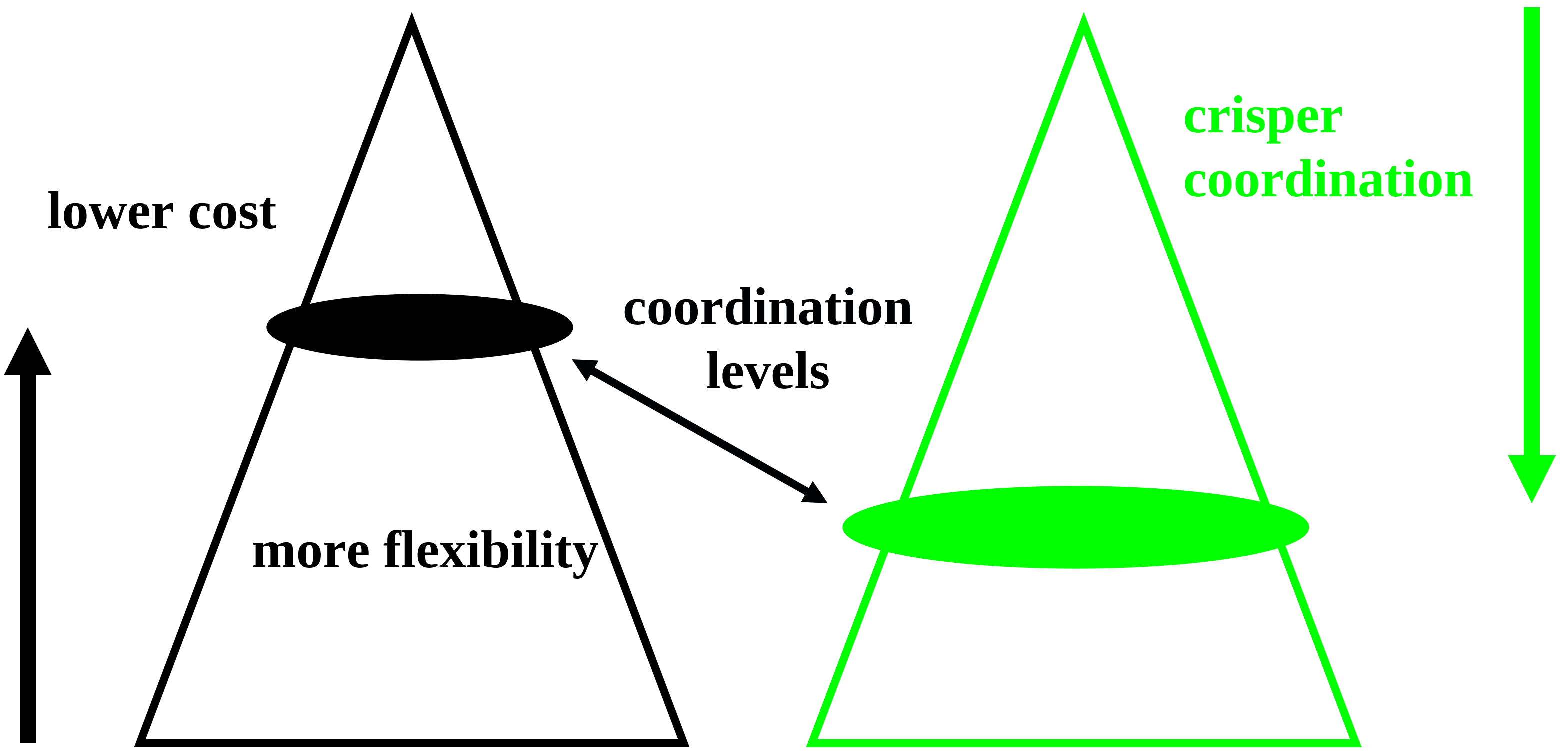
- plans must be fully detailed
- information does not merge when summarized
- OR branch choices do not matter

# Plan Step Merging
## (Yang, 1997; Cox & Durfee, 2003)



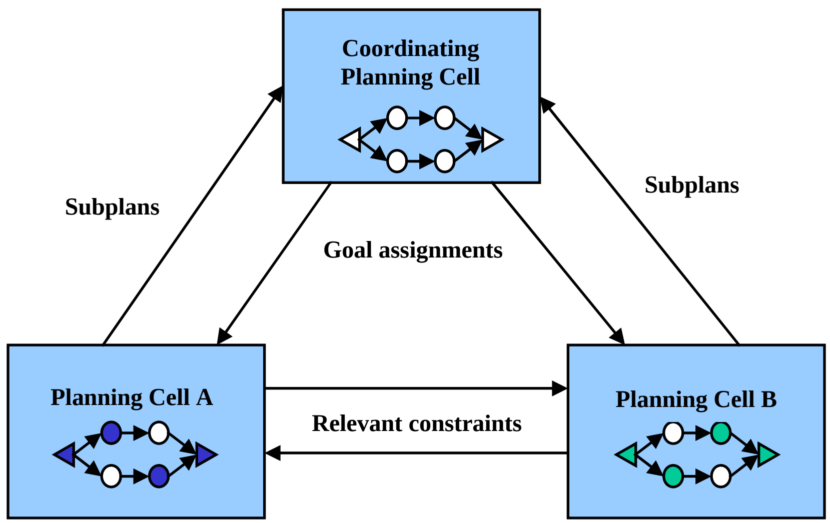
# Tradeoffs

Choice of level at which coordination commitments are made matters!



# DSIPE
## (desJardins & Wolverton 1999)

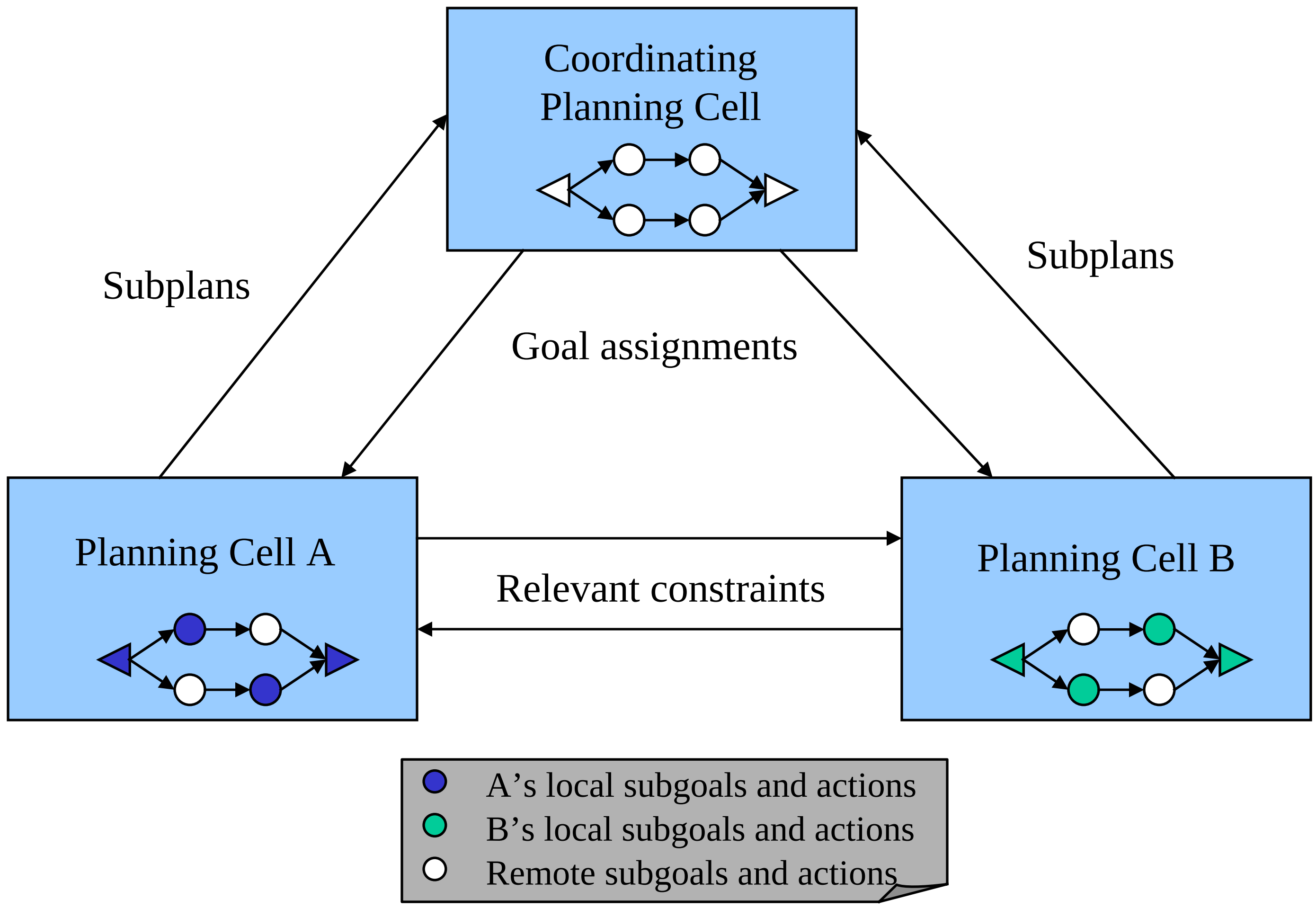
- Distributed version of SIPE-2 planning system
- SIPE – mixed-initiative hierarchical (HTN) planning
- Centralized conflict resolution
- Creates common partial views of subplan
- Synchronization and plan-merging
- Irrelevance reasoning on pre-conditions and effects to limit communication



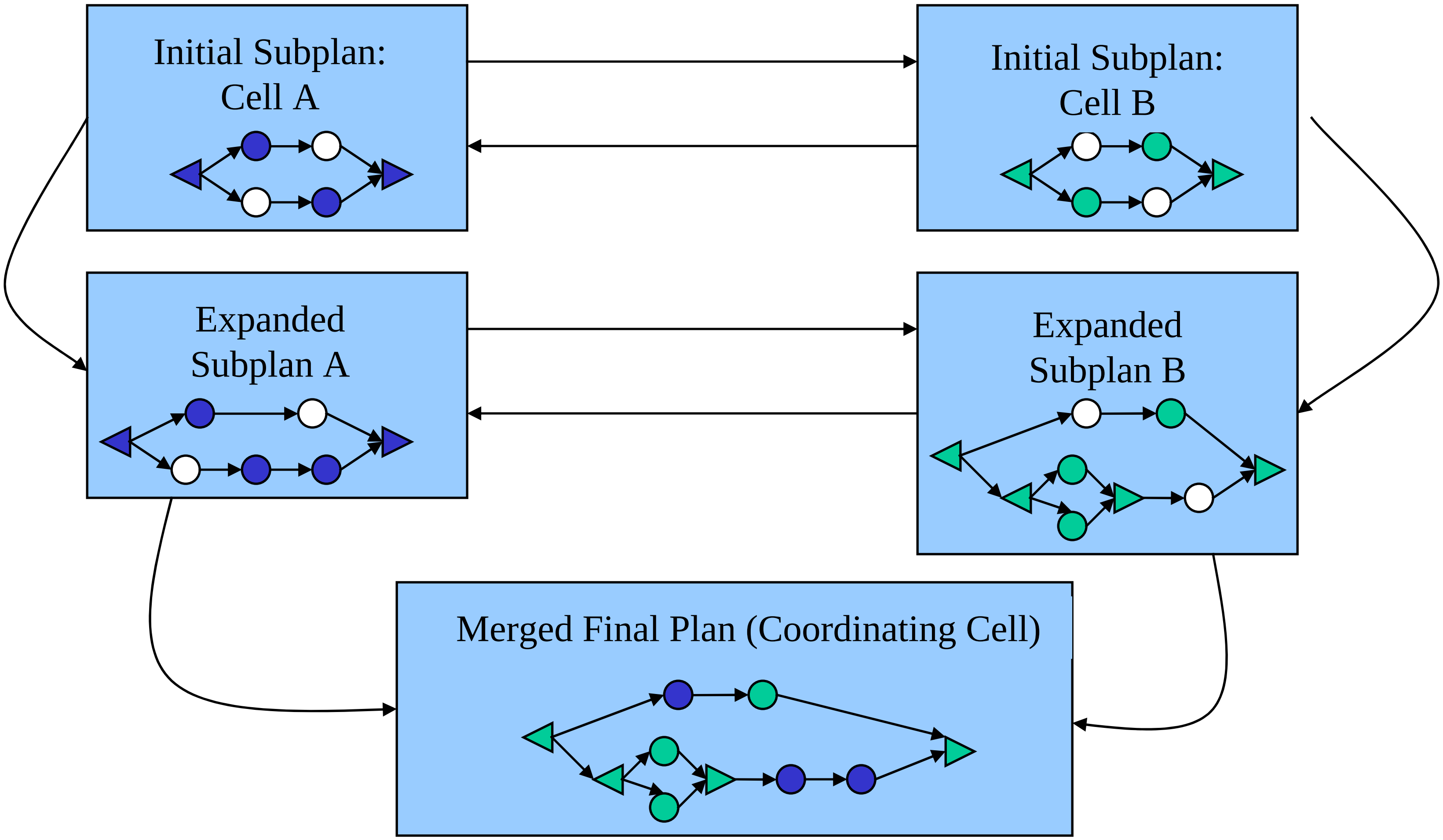
# DSIPE – Motivation

- Distributed planning support
  - Multiple agents developing portions of a joint plan independently
  - Roles of agents are fixed, or constrained by capabilities and experience
  - Each agent communicates planning decisions (effects, resources, etc.) to other agents at planning time
  - Resolve conflicts as they occur: Avoid expensive replanning
- Key issues:
  - Sharing plan structure to aid communication and merging
  - Incremental sharing of information about evolving plan
  - Limiting communication to the most relevant information

# DSIPE Architecture

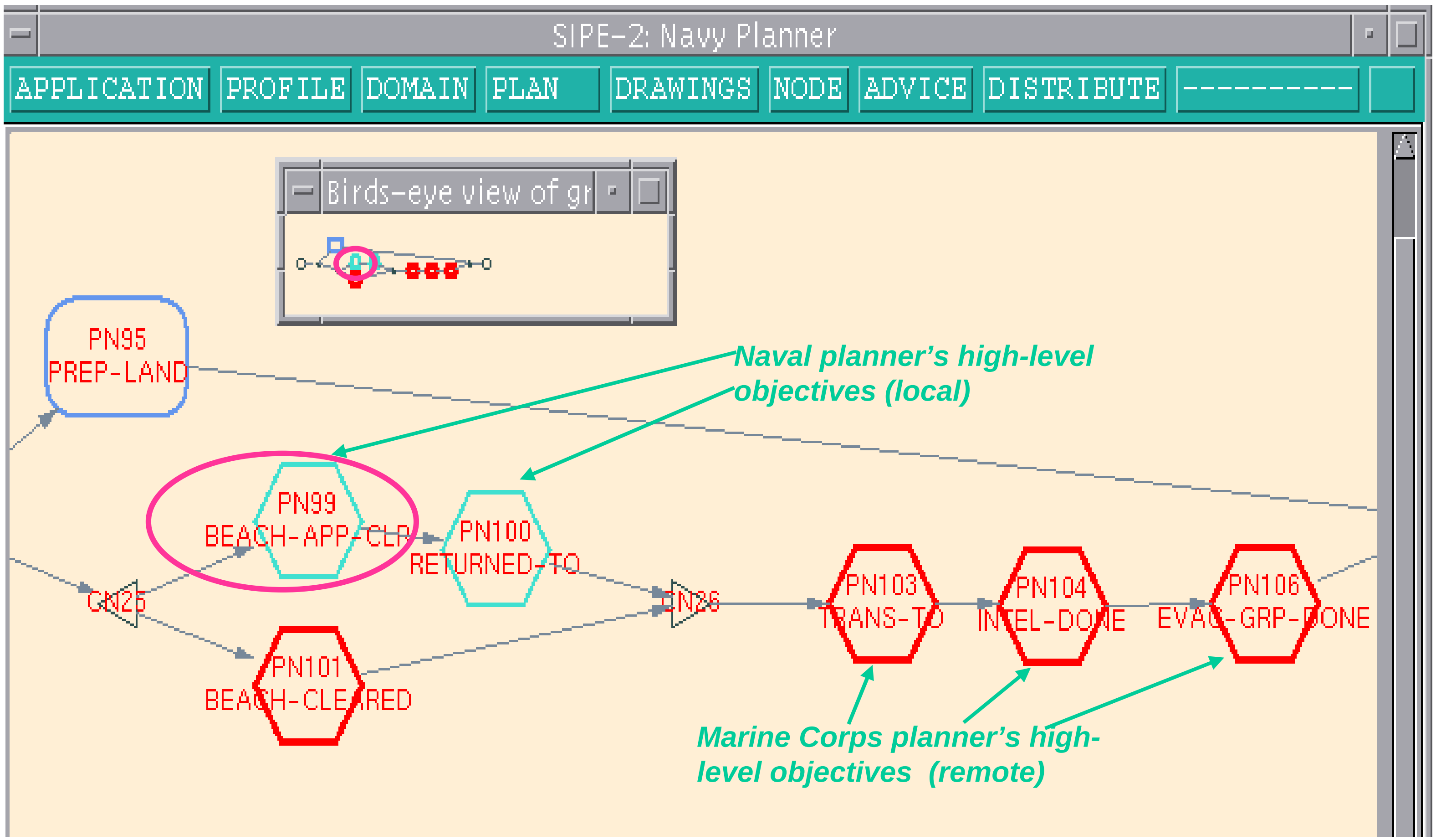


# DSIPE – Approach



# DSIPE – Distributing Goals

- Coordinating cell assigns goals (objectives) to lower-level planning cells
- Local goals/actions shown in blue; remote goals/actions in red

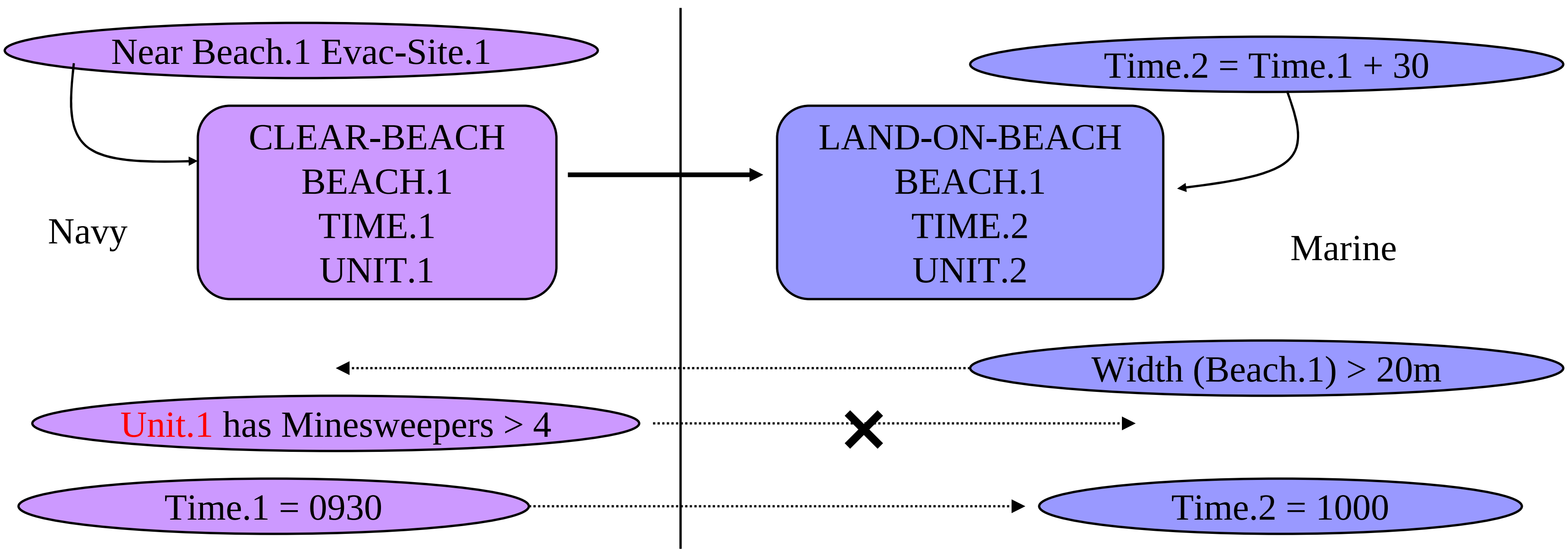


# DSIPE – Common Plan View

- Each plan sees its own plan and “relevant” parts of other plans (“skeletal subplan”)
- Remote objectives serve as placeholders for attaching constraints associated with remote plans
- Shared constraint types include:
  - postconditions (effects of actions)
  - ordering constraints
  - variable constraints (including temporal variables)
  - preconditions, subgoals

# Filtering Variable Constraints

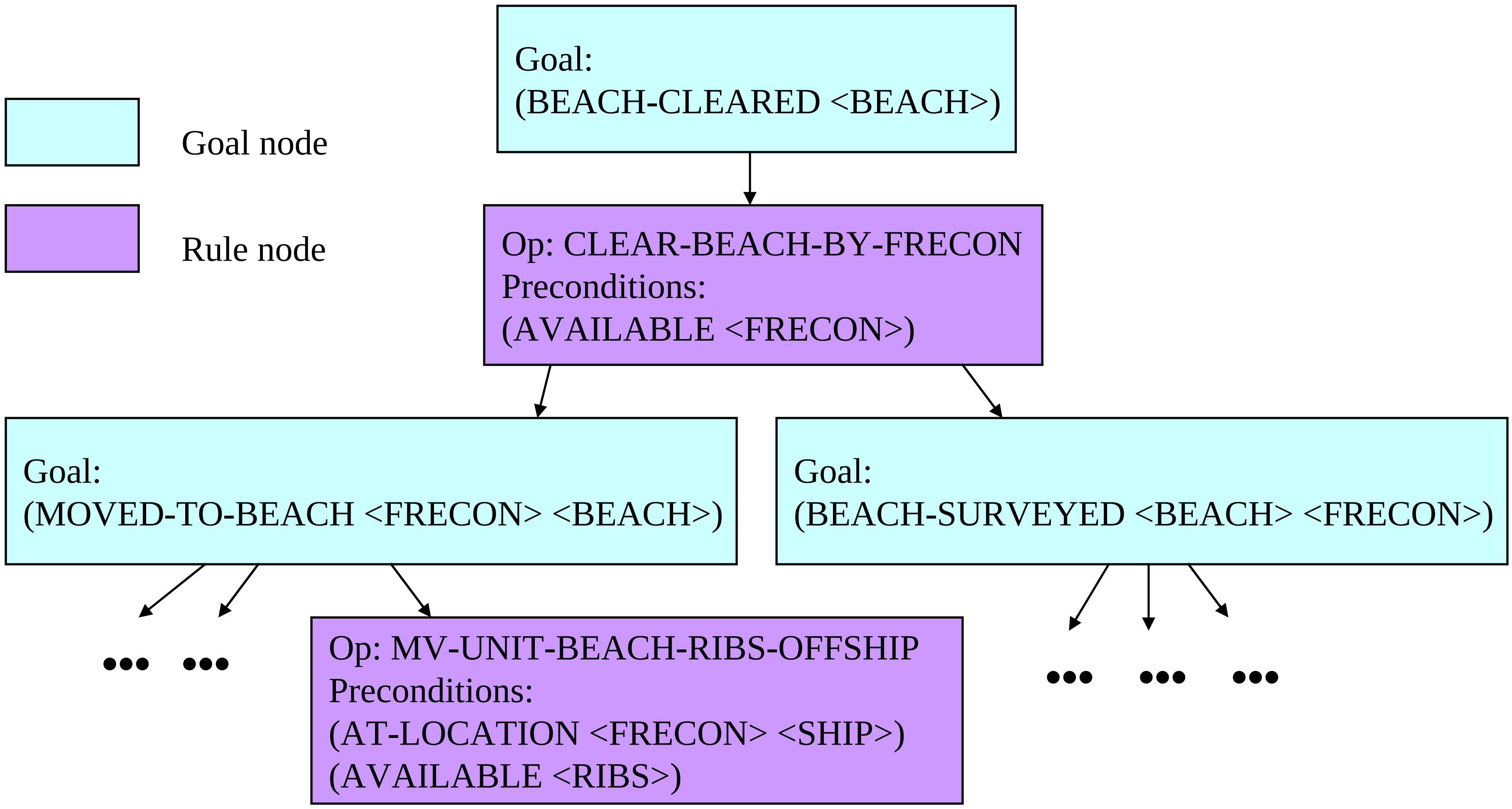
- Shared plan structure gives set of shared variables at the boundary between the subplans
- Share constraints that affect *only* these variables



# DSIPE - Filtering Postconditions

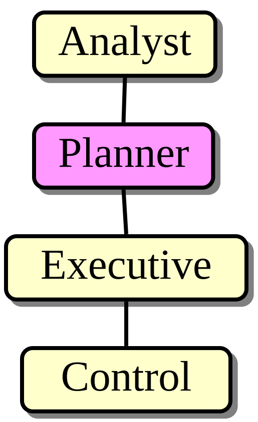
- Use *irrelevance reasoning* to filter out irrelevant messages
- Build *query tree* based on each planning agent's assigned goals and planning operators
- Query tree: facts that are "reachable" by planner
- Send only those planning effects that match query tree
- Using similar approach for preconditions/subgoals

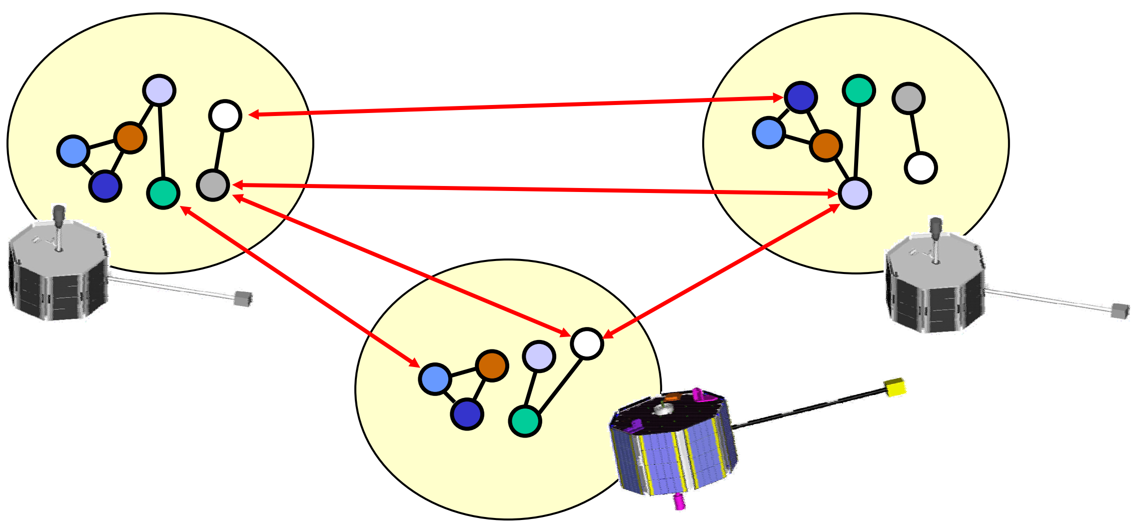
# DSIPE – Query Tree



# Distributed Constrained Optimization

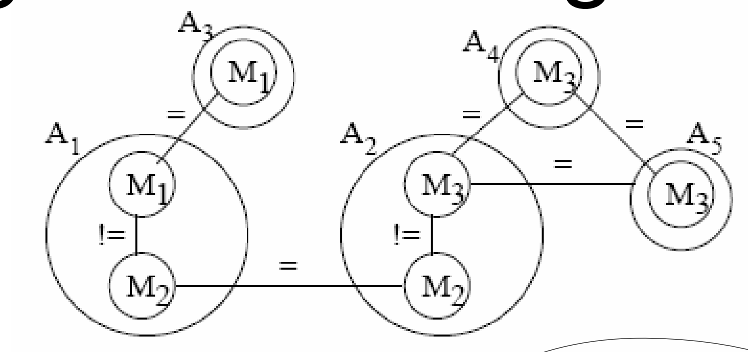
- Optimize a function of variable assignments with both local and non-local constraints.

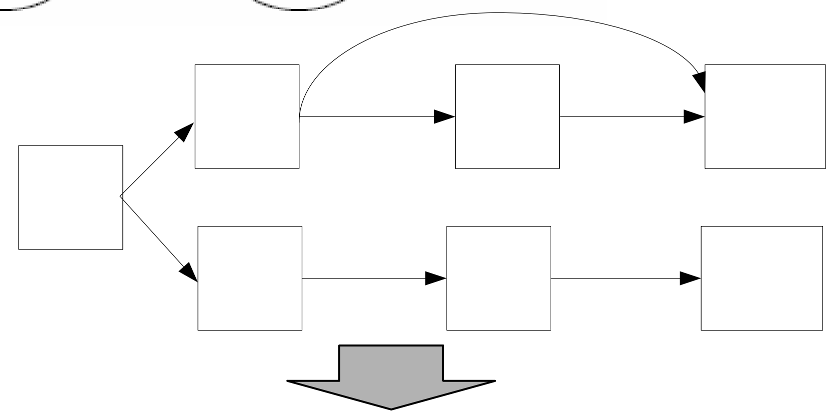


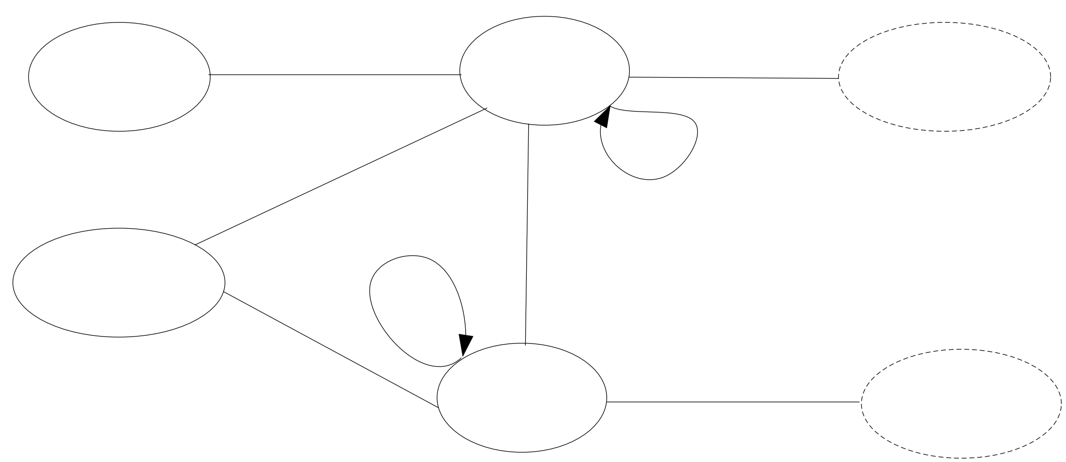



# Distributed Constraint Reasoning for Planning & Scheduling

- DCR = DCSP and DCOP
- Allocating events/resources to time slots (meeting scheduling)
  - Hannebauer and Mueller, AAMAS 2001
  - Maheswaran *et al.*, AAMAS 2004
  - Modi & Veloso, AAMAS 2005
- Coordinating plans by making coordination decisions variables
  - Cox *et al.*, AAMAS 2005





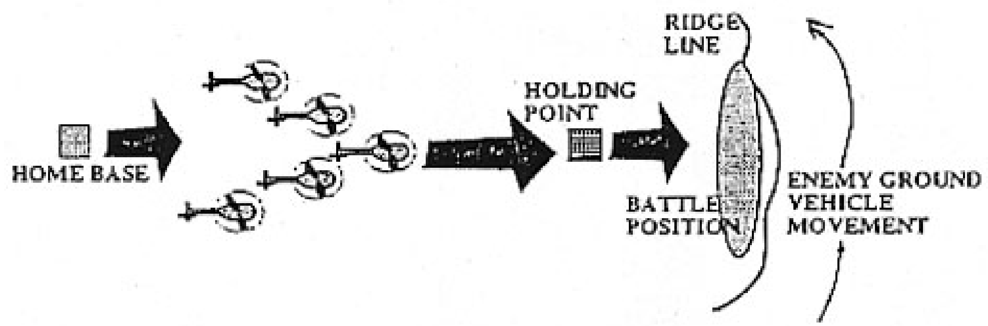
# Market Mechanisms

- Mostly used for resource/task allocation
- Plans share resources and tasks over time (another resource)
- Combinatorial auctions for bids over multiple resources
  – optimization techniques capture constraints and produce schedules
  – if during execution, auction/optimization may need to be repeated for unexpected events
  – difficult to motivate truthful bids and obtain optimal allocations, but no other technique gives such guarantees

# Working Together

- Multi-Agent Plan execution semantics
  - Shared Plans
  - Joint Intentions
    - GRATE
    - STEAM

# Explicit/Procedural Plan Coordination (without underlying semantics)




- Provide specific plans to coordinate
  - When at holding point, scout flies to battle position then informs those waiting at holding point that the battle position is scouted
  - To fly in formation, each agent is assigned a partner to follow in formation
- Difficult to get it right for all contingencies

# Problems with Explicit, semantics-free Plans

*No framework to anticipate failures; numerous ad-hoc plans*

- Upon reaching the holding area, the company waited, while the scout started flying forward.
  - Unfortunately, the scout unexpectedly crashed into a hillside. Hence, the rest of the company just waited indefinitely at the holding area, waiting to receive a message from the (crashed) scout that the battle position was scouted.

- Upon recognizing that the mission was completed, one company member (the commander) returned to home base, abandoning others at the battle position
  - The commander's "partner" agent was unexpectedly shot down, and hence it failed to coordinate with others in its company.

# Joint-Intentions as Mechanism for Building More Complex & Robust Coordination Plans

- Team goals/plans are represented explicitly
- Team members' have commitments and responsibilities toward others when executing a team activity.
  - Commitments to not just local actions, but *achievement of overall goal*

# Explicit Model of Teamwork using Joint-Intentions (Cohen & Levesque)

- A team $\theta$ jointly intends a team action
  - if team members are jointly committed to completing that team action
  - while mutually believing that they were doing it

- Joint commitment is defined as joint persistent goal (JPG)
  - JPG ($\theta$,*p*) where *p* stands for the completion of the goal
  - entire team can be treated as jointly committed to a team plan
    - when company of helicopters flies to a waypoint, each individual is not flying on its own to waypoint while merely coordinating with others.
    - **Success of the team may not require each individual to successfully complete its journey**

# Dissolution of JPG

- JPG ($\theta$, *p*) is dissolved when a team member $\mu$ privately believes that *p* is either achieved, unachievable or irrelevant

- $\mu$ is left with a commitment to have this belief become a mutual belief of all team members
  - relate to breakdown 2

- JPG ($\theta$, *p*, q) includes a common escape clause q

# Conditions for JPG to Hold

- All team members mutually believe that *p* is currently false.
- All team members mutually know that they want *p* to be eventually true
- All team members mutually believe that until *p* is mutually known to be achieved, unachievable or irrelevant, they mutually believe that they each hold *p* as a weak goal (WG)
  - Having privately discovered *p* to be achieved, unachievable or irrelevant, $\mu$ has committed to having this private belief become $\theta$'s mutual belief

# Further JPG Implications

- *Critical expertise heuristic*: If the success of the team's joint intention is solely dependent on the role of an individual agent, then the agent's role non-performance (failure) implies that the team's joint intention is unachievable.

- *Dependency heuristic*: If an agent's own role performance is dependent on the role of the non-performing agent, then the agent's own role performance is unachievable.

# Castelfranchi's Counter-example (ICMAS'95)

- Two scientist, one French (F) and one American (A)
- Both searching for the AIDS vaccine
- Mutual beliefs among F and A that searching for AIDS vaccine
- Both have identical goals
- Both will let each other know if AIDS vaccine located
- However, not a JPG, because they compete with each other?

# SharedPlans (Grosz & Kraus, 96)

- No joint mental attitude, instead “intention that” for helpful behavior
  - Guides an agent to take actions, including communicative actions, that enable or facilitate its teammates to perform their assigned tasks
  - Joint Intention really only addresses when team problem solving can begin and how it should be terminated when certain conditions occur

- SharedPlan of a group GR requires that:
  - Mutual belief that each member intends that GR achieves joint goal
  - Mutual belief (agreement) in the joint recipe
  - For each step in the recipe:
    - Some individual/subteam forms SharedPlan for that step
    - Other members believe their exists a recipe for that SharedPlan
    - Other members **“intend that ”** individual/subteam perform the step

- SharedPlans may be partial (e.g., recipe not fully elaborated)
  - Entire web of intentions and beliefs for teamwork defined

***More Sophisticated View of the Semantics of Teamwork***

# Defining Intention in Shared Plans

- Intention-To (G, $\alpha$, $T_i$, $T_a$, $C_\beta$) represents agent G's intention at time $T_i$ to do action $\alpha$ at time $T_a$ in the context $C_\beta$ (higher level plan)

- Intention-That (G, *prop*, $T_i$, $T_{prop}$, $C_{prop}$) represents agent G's intention at time $T_i$ that a certain proposition *prop* hold at time $T_{prop}$ in the context of $C_{prop}$
  - Prop -- There exists some individual or subgroup to do a task which is part of the recipe for the SharedPlan

# SharedPlans (Cont)

- Intention.That (int.th) core concept defined via several axioms, for example
- Axiom A7:
  - Group GR has a sharedplan S1
  - G1 is a member of GR, G2 is a member of GR
  - G1 has **intends.that** for G2 to bring about some action A2 in service of S1
  - G1 can perform A1
  - ***G1 believes that G1's performing A1 and then G2's performing A2 will be cheaper (lower cost) than G1 not performing A1 and G2 performing A2***

Then G1 will consider performing A1

# Potential Intentions
## Pot.Int.To, Pot.Int.That

- agents’ mental state prior to deliberating about intentions in context of other intentions it holds

- weigh different possible courses of action or options

# Axioms for Helpful Behavior

(INT2) Axiom :

*G* Int.Th *some prop that G believes isn't true*

$$\text{Int.Th}(G, prop, T_i, T_{prop}, C_{prop}) \wedge \text{BEL}(G, \neg(prop), T_i) \wedge$$

*G believes it can do something (α) to help*

$$\text{BEL}(G, (\exists \alpha, R_\alpha, T_\alpha)\ \text{Result}(\text{Do}(G, \alpha, T_a, \text{constr}(C_{prop})), prop), T_i) \wedge$$

$$\text{BEL}(G, \text{CBA}(G, \alpha, T_a, R_\alpha, \text{constr}(C_{prop})), T_i) \quad \Rightarrow$$

*G will consider doing α*

$$\text{Pot.Int.To}(G, \alpha, T_i, T_a, C_{\alpha/prop})$$

Axiom (INT2) states that if an agent has an intention-that toward some proposition that it believes does not currently hold and the agent believes it is able to do some act α that will bring about the proposition's holding, then the agent will consider doing α. The potential intention to do α will cause deliberation about adopting an intention to do it, and, barring conflicts, lead to this becoming a full-fledged intention.

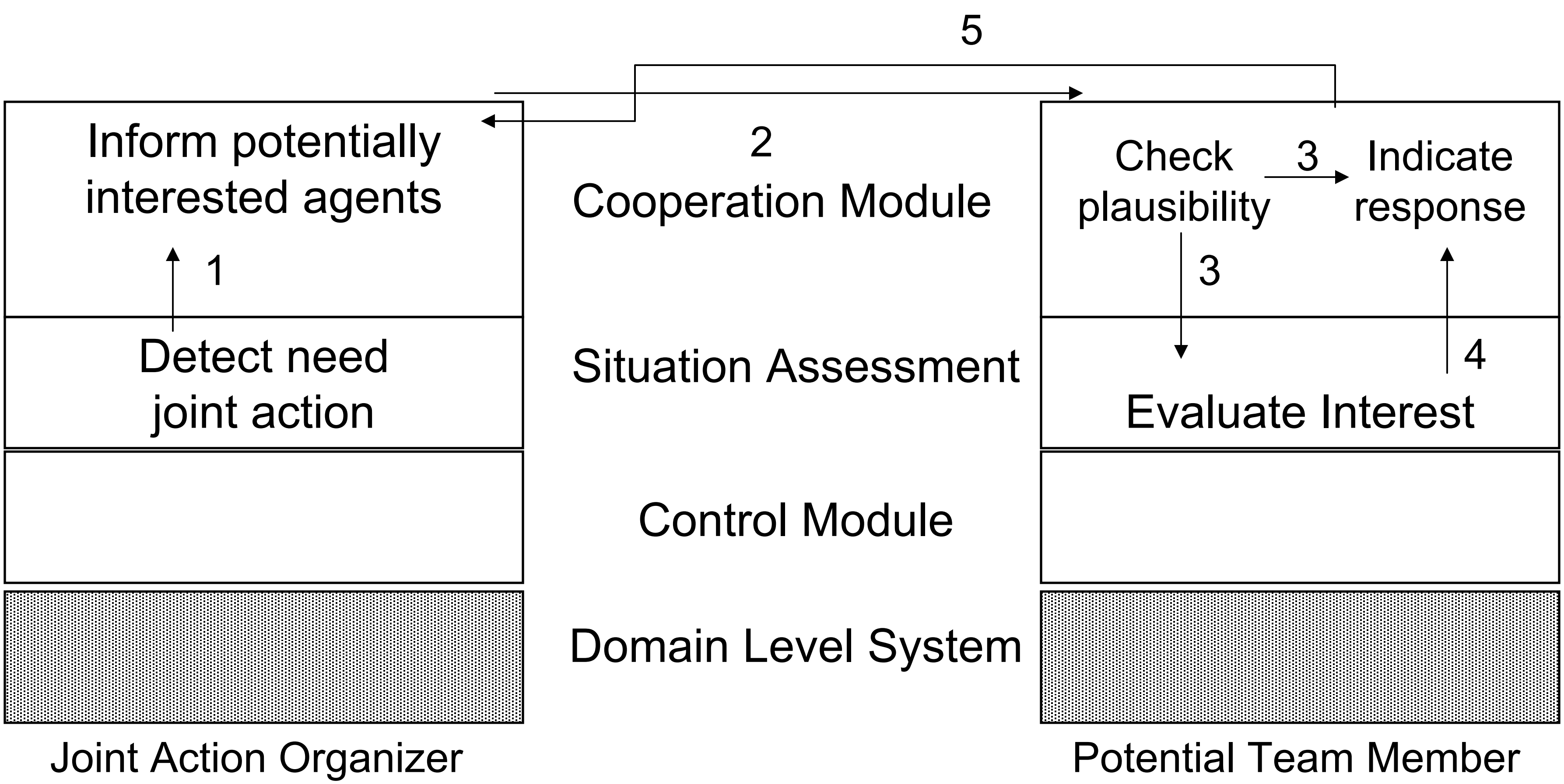
# Practical Teamwork -- GRATE(Jennings) & STEAM (Tambe)

- Computationally tractable versions of Joint-Intention Semantics
  - Reasoning from first principles in modal logic very expensive
- Need Framework for Specifying Plans and Recognizing Existence of Joint Goals
- Still multi-agent plan execution (some “recipe” must already have been created)

# GRATE*
## *(Jennings)*

- Monitor local events and environmental changes
- Create new objectives
- *Plan for achieving new objectives*
  - means-ends analysis
  - recipe library
- Determine whether need help
  - can it be done locally given current intention (compatibility checker)
  - if inconsistency, attempt to either modify existing commitment or alter the objective to remove conflict
    - based on agent's preferences
    - if need to do collaboratively
  - identify agents who are able to help
  - a skeletal joint intention is constructed
  - construct final team and joint plan
    - individual agents verify they can accomplish subplan

# Establishing a Joint Action



# GRATE* Distributed planning protocol

**PHASE 1**

Organizer detects need for joint action to achieve goal *G*.

Organizer contacts all acquaintances capable of contributing to *R* to determine if they will participate in the joint action using the Responsibility cooperation model.

Let: Ω = set of willing acquaintances.

**PHASE 2**

*FORALL* actions in *R*

select agent A ∈ Ω to carry out action θ ∈ *R*

(criteria: minimize number group members)

calculate time ($t_\theta$) for θ to be performed based on temporal orderings of *R* and the anticipated communication delay

send (θ, $t_\theta$) proposal to A

A evaluates proposal against existing commitments (C's)

IF no-conflict (θ, $t_\theta$) THEN create commitment $C_\theta$ for A to (θ, $t_\theta$)

IF conflicts ((θ, $t_\theta$), C) ^ priority (θ) > priority (C)

THEN create commitment $C_\theta$ for A to (θ, $t_\theta$) and re-schedule C

IF conflicts ((θ,$t_\theta$), C) ^ priority (θ) < priority (C)

THEN find free time ($t_\theta + \Delta_\theta$), note commitment $C_\theta$ and return updated time to leader

Return acceptance or modified time to team organizer

IF time proposal modified THEN update remaining actions times by $\Delta t_\theta$

*END-FORALL*

# STEAM: A Shell for TEAMWORK [Tambe]

- Extends Joint-Intention Framework to handle
  - Communication costs
  - Uncertainty about state other team members
  - Single and Multiple Team member  failure
  - Partial Satisfaction of Goal
  - Evolving hierarchy of joint events

- Introduces more bottom-up approach to establishing joint-intentions

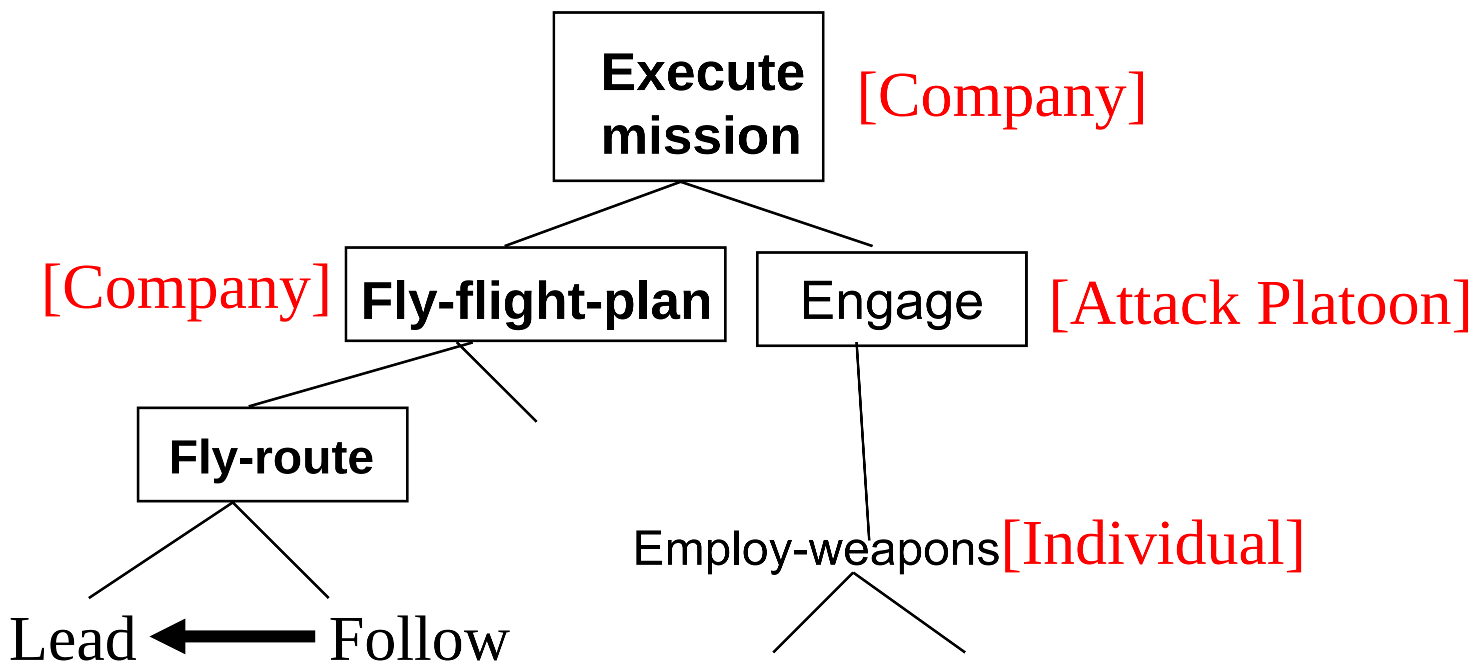
- Organizational roles
  - Role dependency

- Implements Joint-Intention Framework in  Environments
  - where agents are interacting with the world
  - where they can sense the state of the world
  - observe actions of other agents

# STEAM Overview

Team-oriented Programs: Explicit team reactive plans

- Hierarchically expand into individual/subteam plans
  - *Each plan has preconditions, body, termination conditions*
- Roles, e.g., *lead role* in formation flying, with constraints
- Assign teams/subteam to team plans based on capability



# Situated Plans (Reactive Plans)

- Situated/reactive plan consists of:
  - *Preconditions*, matched with agents' beliefs
  - *Termination conditions*, to terminate plan when matched
  - *Plan body* to execute when plan activated
    - May invoke external or internal or no action

- Example: Plan Attend-Agents-Workshop
  - Precondition: *Saw agents workshop call for participation*
  - Body: *Register for workshop, fly, attend sessions, fly-back..*
  - Termination condition*: Attended agents workshop*

# Establish Commitments Protocol

1. Team leader broadcasts a message to the team $\Theta$ to establish PWAG (persistent weak goal achievement) to operator OP. Leader now establishes PWAG. If $[OP]_\Theta$ not established within time limit, repeat broadcast.
2. Subordinates $\nu i$ in the team wait until they receive leader's message. Then, turn by turn, broadcast to $\Theta$ establishment of PWAG for OP; and establish PWAG.
3. Wait until $\forall$ $\nu i$, $\nu i$ establish PWAG for OP; establish $[OP]_\Theta$

# STEAM Overview
## Team Plan Execution: Communication

*<u>All team plans executed by forming & terminating joint commitments:</u>*

- **Request-confirm** exchanges so all team members select & commit
- Establish mutual belief for achieved, unachievable,… to terminate
- *Forming & terminate team plans: All communication in STEAM*

*<u>Example: Team of helicopters jointly commit to “execute mission”</u>*

- If commander privately believes *mission unachievable…*
- Commander must establish mutual belief in termination condition
- It communicates *mission unachievable*: no one left behind

*<u>Hierarchy of jointly committed team plans and subteam plans:</u>*

- *<u>Team coherent when executing & terminating team plans</u>*

# Implementing Models of Teamwork
## STEAM Overview: Monitor and Repair

Addresses unanticipated team member or subteam failure:

- Monitoring & replanning capabilities
  - Explicit constraints individual/subteam roles & team goal
    - *AND: All roles must be fulfilled*
    - *OR: At least one rule must be fulfilled*
    - *Role-dependency →: Role R1 dependent on R2*
  - Constraints may be combined, e.g., ((A OR B) AND (B → C)

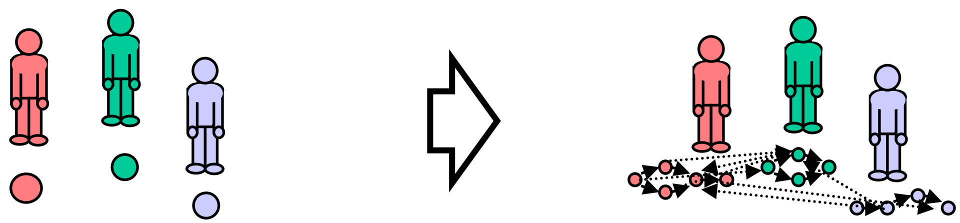
- Scouting failure example:
  - *Wait-for-battle-position-scouted is the team plan*
    - *AND-combination*: Scout and Non-scout roles in team plan
    - If scout crashes, the scout role is not fulfilled
    - AND-combination implies that the team plan fails

# STEAM Overview (Continued)

- Joint commitment to replan by reorganization, if critical failures
  - Determine candidates for roles via capability matching
  - Candidates for roles ensure no conflicting critical commitments
  - Individual/subteam may volunteer
    - *If multiple candidates, compare based on capability*
  - Highest capability agent wins

- Scouting failure example continued
  - *Wait-for-battle-position-scouted is the failed team plan*
    - Locate other pilots capable of scouting
    - New candidate scout ensure no conflicting commitments
    - Candidate scout(s) volunteer
    - Best capability scout wins

# Distributed Continual Planning

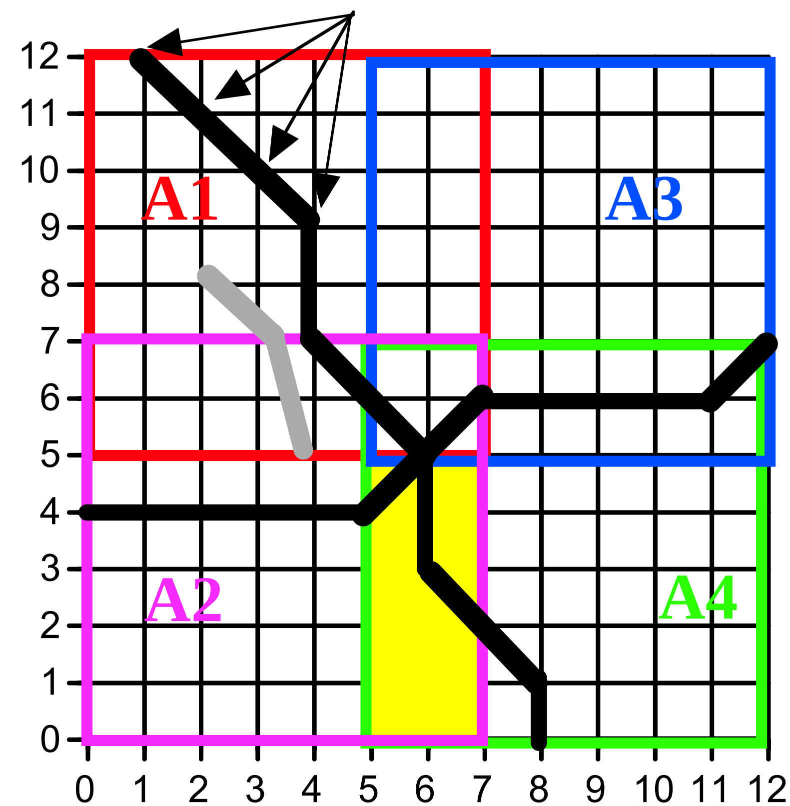


- Same as prior case (distributed planning), but
  - plans are being executed at same time
  - goals may change
- At any given time, plans might only be partially coordinated, and execution results could cause chain reactions of further planning and coordination
- May break and re-make commitments
  - unexpected event/failure
  - goal change
- Must reach consensus (and deconflict) on plan segments before they are executed
  - real time guarantees?
  - what if not possible?
- In a sense, the coordinated plans are only evident after the fact, as they are continually being adjusted during execution

*[1981–1991]*

# The Distributed Vehicle Monitoring Problem

- Acoustic vehicle tracking
  - grammar specifies vehicle's "signature"
  - varying signal strengths
  - uncorrelated noise
  - "ghost tracks"
- Multiple agents with overlapping sensors
  - faulty sensors
- Coordinate processing to terminate as quickly as possible

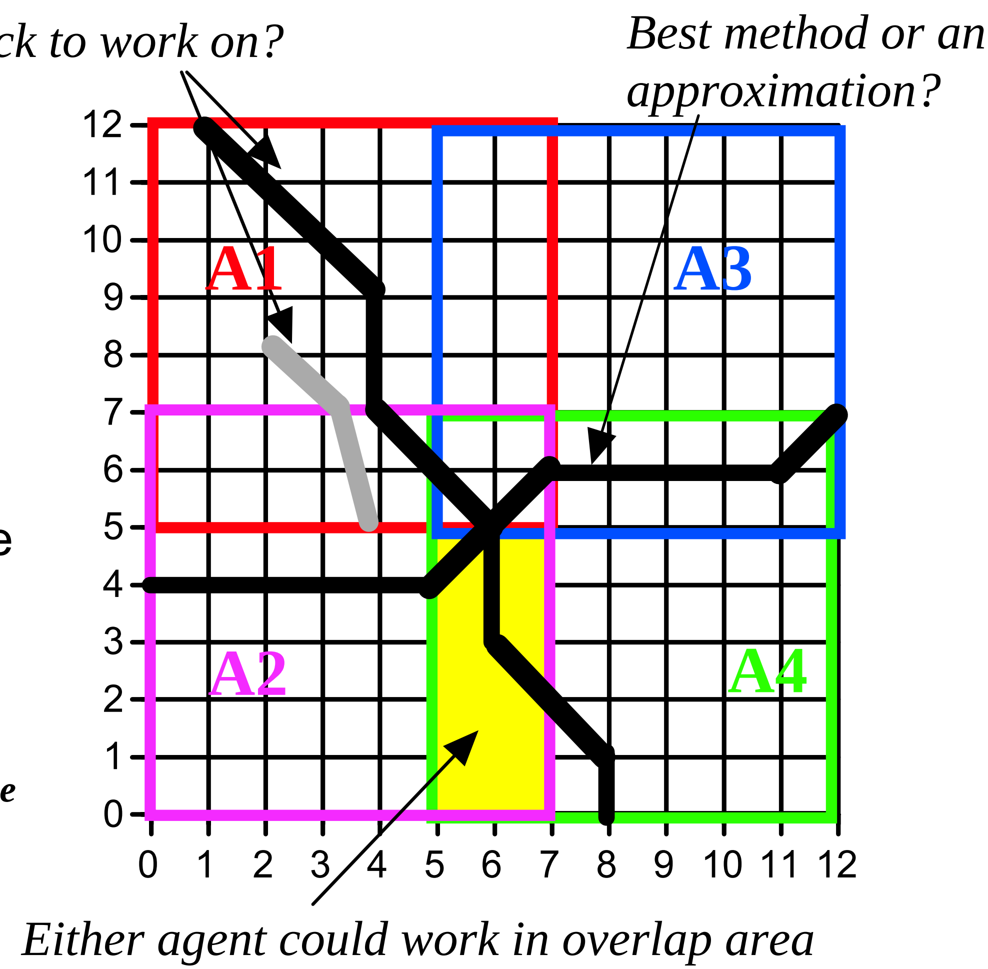


# The Distributed Vehicle Monitoring Problem

*[1981–1991]*

- Making choices about what activity to do...
  - ...in what order
  - ...and at what time
- Coordinate processing to terminate as quickly as possible
- These choices do make a difference

***Note: in this problem we assume agent cooperation***



# Distributed Continual Planning via Local Plan Merging

*By combining together interacting local goals/plans of different agents, an agent constructs **partial global goals and plans***

– To guide an agent in reordering its actions so as to exploit results from other agents and avoid resource contention
– To provide in a timely manner results that could be helpful for the solution of other agents goals
– To avoid the redundant solution of goals except where desirable
– To achieve a more accurate view of the global importance of it achieving a local goal

# Partial Global Planning
## (Durfee & Lesser, 1991)

Each agent constructs and maintains an intermediate level view of its likely plans that would occur over the near term.

- Expected order it would pursue its goals
- Estimates of the time required to solve each goal
- Importance of the goal and the quality of the expected result
- High level plan for locally solving each goal

- *Use meta-level organization to know who is responsible for what aspects of plan coordination—to whom to send this info*

# Recognizing More Global Goals

- Each agent receives subset of *other* agents' goals and plans
  - Subset leads to **partial global view**
  - Potential for different agents having different views
- Reduce computational and communication costs by transmitting only "best" goals/plans
  - update model of other agent activities only when their plans change.
- Compare goals of different agents' plans:
  - use simplified domain knowledge,
  - find goals that could be part of a larger goal,
  - generate partial-global-goal (PGG).

# Improving Coordination

- PGP interleaves participants' planned activities into *plan-activity-map*:
  - each activity has predicted start and end times, results
  - plan-activity-map roughly predicts concurrent activities
- Rates each activity based on expected costs and results, how it is affected by preceding acts, and how it affects succeeding acts
- Iteratively reorders acts until sum of ratings does not improve
  - hill-climbing, possibly non-optimal ordering

# Planning Solution Integration

- Identify when each piece will be developed at a problem solver;
- Iteratively find earliest time and location where pairs of results can be combined and form solution-construction-graph ;
- Permit integration redundancy to increase reliability.

# Issues in Solution Construction

- Graph improves communication decisions by only sending information when needed;

- Graph improves flexibility (time-windows) for choosing plans to pursue.

  – Introduces expected interactions -- primitive form of commitment: current decisions based on assumed future activity;

  – change of plans causes retractions of assumptions.

# Partial Global Planning (cont)

- Mapping back to local plans: Partial global plan commitments are internalized
- Local plan execution

- Cycle repeats as local plans change or new plans from other agents arrive.  Always acting on local information means that there could be inconsistencies in global view, but these are tolerated

# Key Assumptions of PGP

- Agents can predict the intermediate-level goal structure that is the focus of their near-term work with some level of accuracy and without significant computation;
- Agents can estimate how long it takes to achieve goals;
- Agents generally follow the prescribed order for achieving goals;
- Agents can recognize the major subproblem/goal interactions among agents using intermediate-level goals;
- Agents can transmit intermediate-level goal structure without significant communication costs.

# Generalized Partial Global Planning

- Domain-independent, coordinated scheduling of agent actions
  - Action choice, order, and timing
- Generalizes and extends Durfee's PGP algorithm, and von Martial's work on task relationships
  - Deadlines
  - Heterogeneous agent capabilities
  - Communicate less info, and at multiple levels of abstraction
- Individual Coordination Mechanisms
  - Recognize certain task structure patterns
  - Re-write the agent's HTN
  - Respond via instantiating a protocol for communicating commitments, non-local task structure information, and partial results.
- Works in conjunction with agent's local task scheduler to remove uncertainty
  - (DTC — Wagner; DTT — Garvey; DRU — Graham)

# TÆMS Task Structure Representation

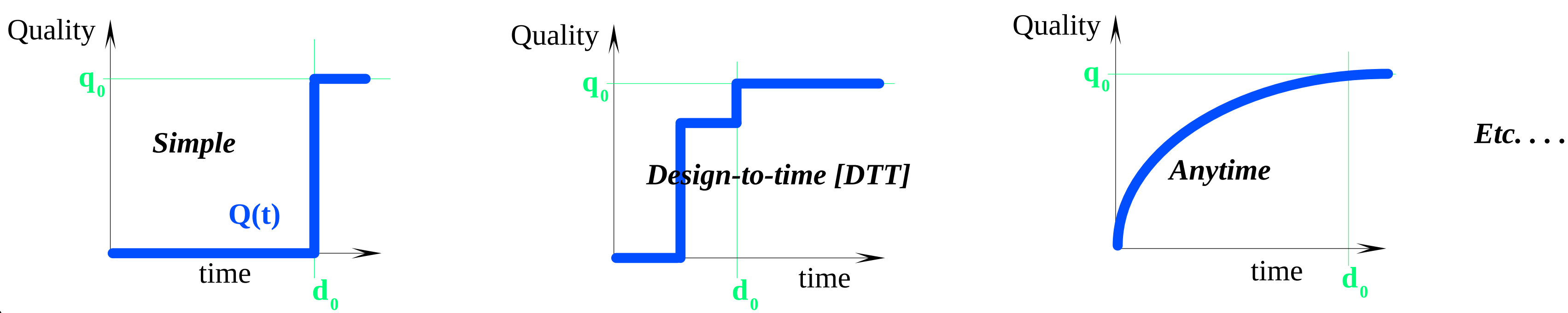
- Representing the “interdependencies” that need to be managed in “complex” domains
  - worth-oriented (vs. state- or task-oriented)
  - time-oriented (synchronization, not just choreography)
  - distributed: no global view
  - uncertainty in action characteristics & outcomes

# TÆMS Task Structure Representation

- "Interdependency" = quantitative change in task characteristics when another task is executed
  - Quality
  - Cost
  - Duration (vs. deadline)
- State-based semantics
- Annotation for HTN style task networks

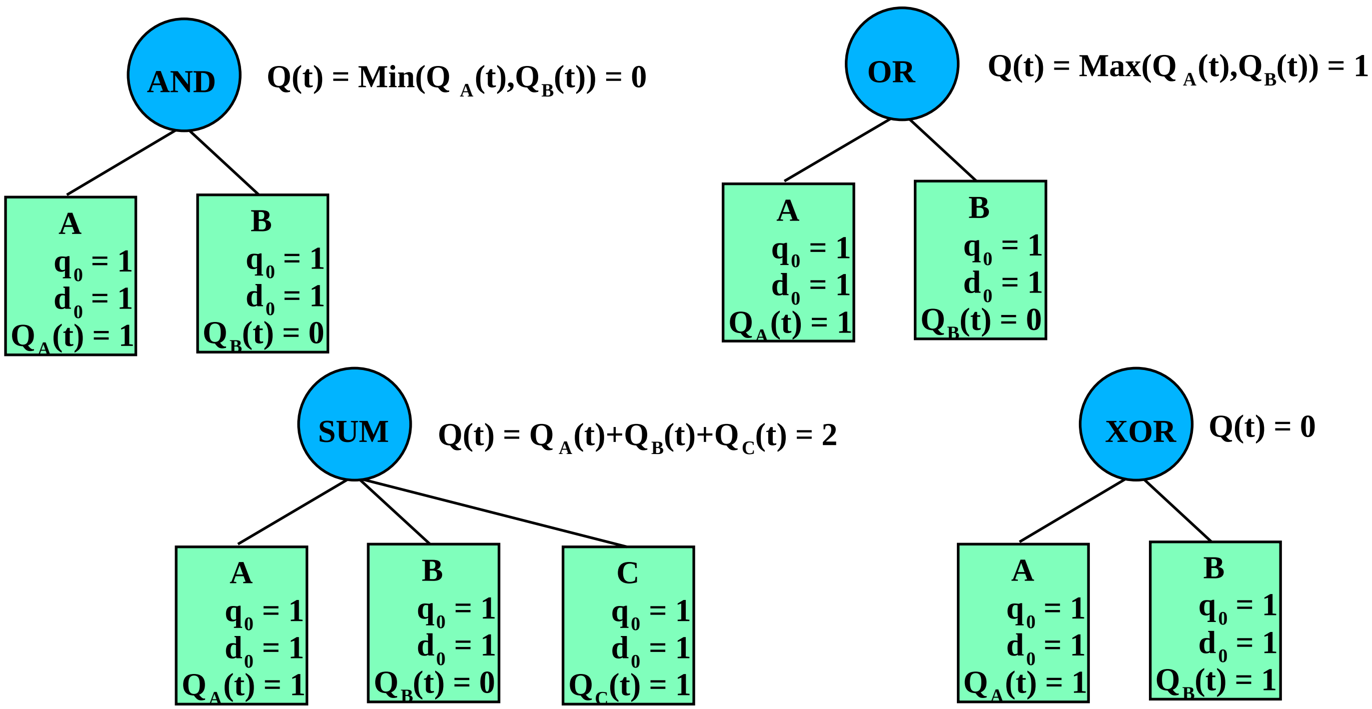
# Actions/Executable Methods

- Characteristic Vector
  - maximum possible cost, quality, duration [$\mathbf{c_0}$, $\mathbf{q_0}$, $\mathbf{d_0}$]
  - associated uncertainty
- Execution Profile
  - start, suspend/resume, finish
- Accumulation Function: Characteristics vs execution time
  - Quality Accumulation Function [QAF]



# Tasks

- Characteristic Accumulation Functions
  - Quality Accumulation Function [QAF]



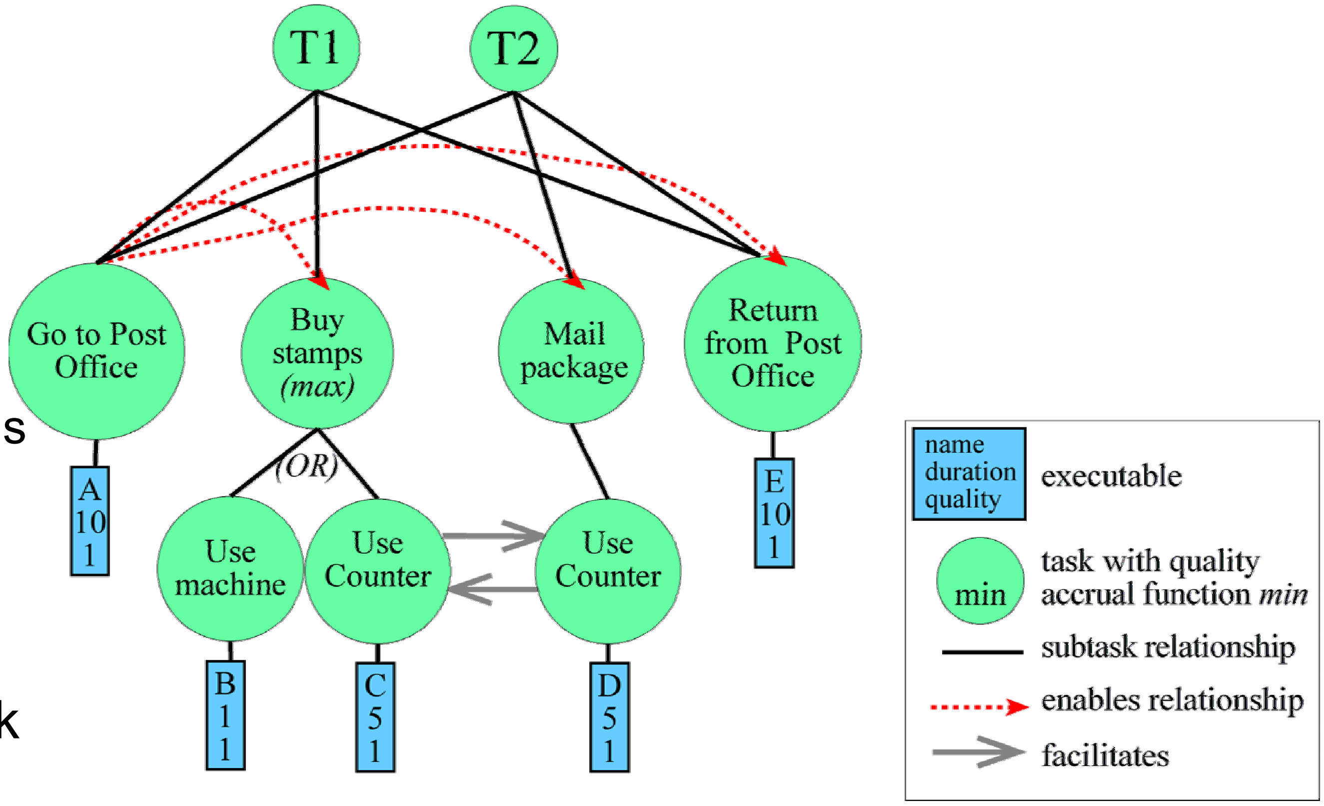
# Performance Measure

- Utility function over characteristic vector
  - – maximize quality
  - – maximize quality - cost
  - – minimize duration subject to $Q_{actual} > Q_{min}$
  - – etc.

# TÆMS Representation Framework

*Develop a representation framework to specify the task structure of any computational environment*

- Performance is: attempt to maximize quality(worth)
- Representation of structure at multiple levels of abstraction
  - Tasks
  - Executable methods
  - Methods have duration, max quality, QAF
- Explicit, Quantitative representation of task interrelationships



# Non-Local Effects & Coordination Relationships

- NLE's are defined when the execution of one method changes the duration or quality or cost of another
- NLE's give an environment its unique characteristics
- A NLE may depend on the communication of information
- A NLE between parts of a task structure known by different agents is called a coordination relationship

# NLEs have quantitative defs

$$enables(T_a, M, t, d, q, \theta) = \begin{cases} [d, 0] & t < Finish(T_a) \\ [d, q_0(M)] & t \geq Finish(T_a) \end{cases}$$

---

$$\begin{aligned} facilitates(T_a, M, t, d, q, \phi_d, \phi_q) &= [d(1 - \phi_d R(T_a, Start(M))), \\ &\quad q(1 + \phi_q R(T_a, Start(M)))] \\ \text{where } R(T, s) &= \frac{Q_{avail}(T, s)}{q_0(T, s)} \end{aligned}$$

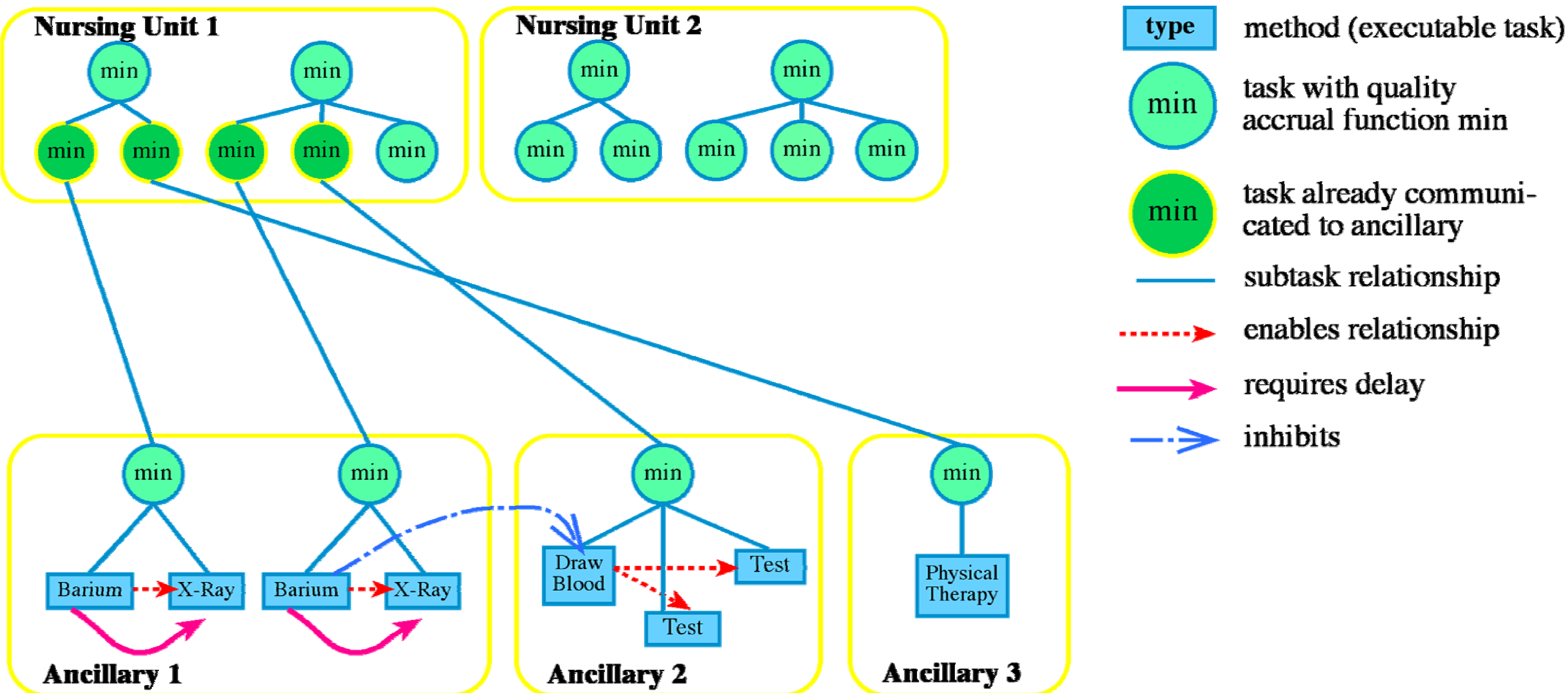
---

$$mutex(T_a, M, t, d, q) = \begin{cases} [d, 0] & Start(T_a) \leq t < Finish(T_a) \\ [d, q_0(M)] & otherwise \end{cases}$$

# TÆMS Usage

- TÆMS can be used for environment modeling, algorithm analysis, and simulation
  - UMass simulators: TÆMS2, MAS
  - DARPA COORDINATORS
  - Agents may use any internal representation; but if task structure is created dynamically must translate
- However, can use TÆMS to build domain independent reasoning capability into an agent architecture that represents task structures internally
  - Planning, Scheduling, Coordination

# Hospital Scheduling

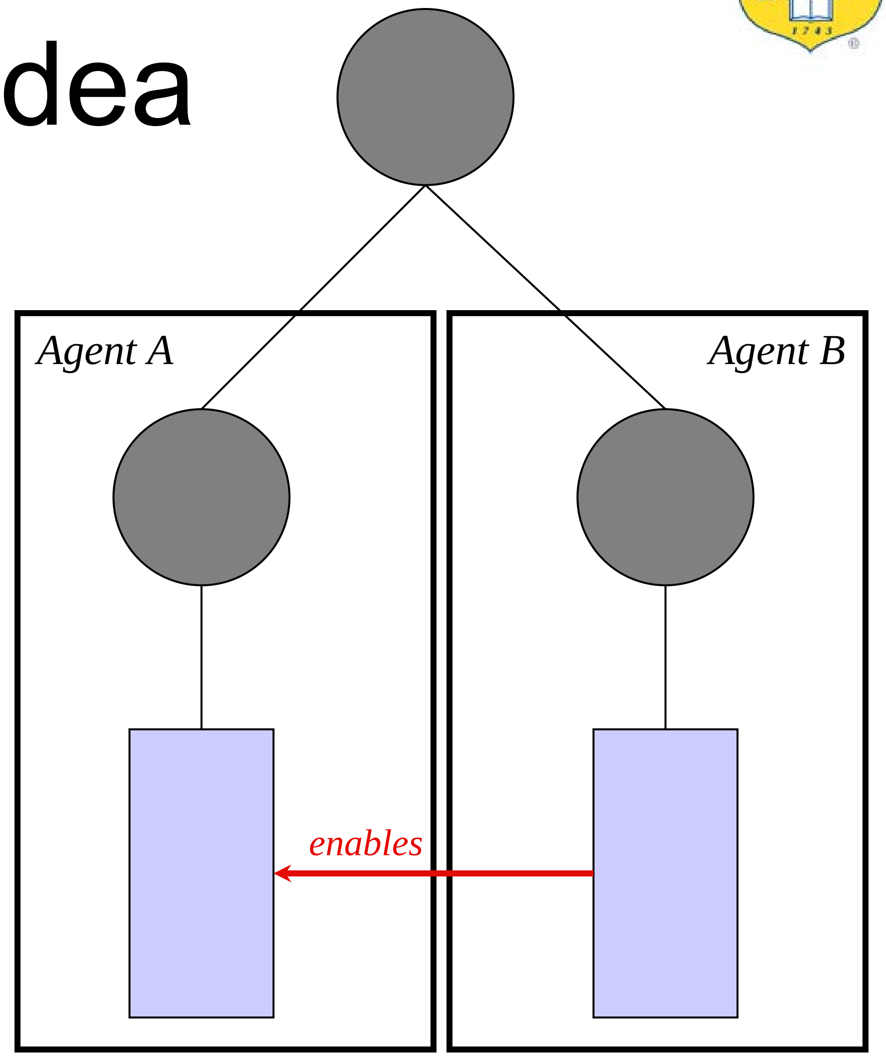


# Generalized Partial Global Planning
## (GPGP, Decker & Lesser, 1995)

- Mechanisms to generalize PGP
  - updating non-local viewpoints
  - communicating results
  - handling redundancy of effort
  - resolve conflicts (hard constraints)
  - handle soft constraints ("optimize")
- Examines tradeoffs of using mechanisms according to
  - communication overhead
  - execution time
  - plan quality
  - missed deadlines

# GPGP: The Idea

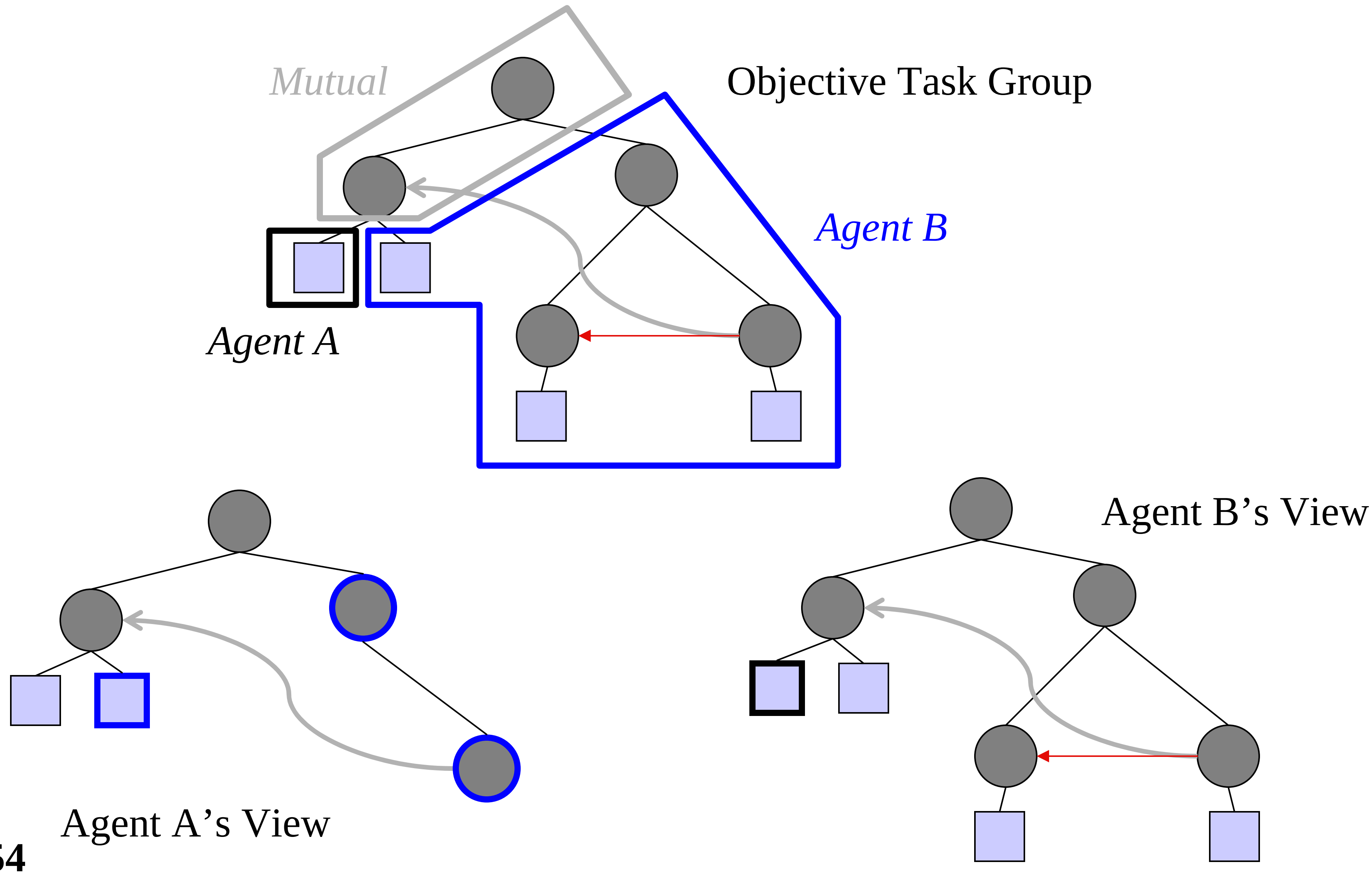
- Have A wait and see (poll)
- Have A ask B
  - "If"
  - "When"
- Have B tell A
  - B sends result when available
  - B commits to a deadline by which it will send the result
- Etc.



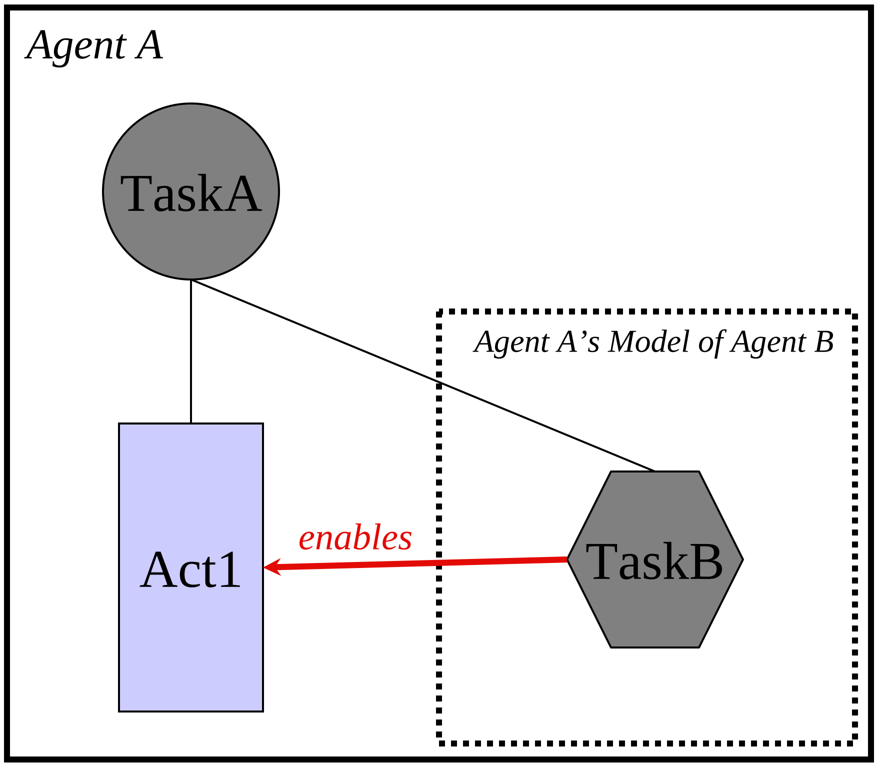
# Some Coordination Mechanisms for Enablement

- Avoidance (with/without quality sacrifice);
- Reservation schemes;
- Simple predecessor-side commitments (to do in future time point, do by deadline, do after EST);
- Simple successor-side commitments;
- Polling approachs (busy querying, timetabling, constant headway);
- Shifting task dependencies by learning or mobile code (promotion/demotion shift);
- More complex multi-stage negotiation strategies;

# Minimizing non-local information

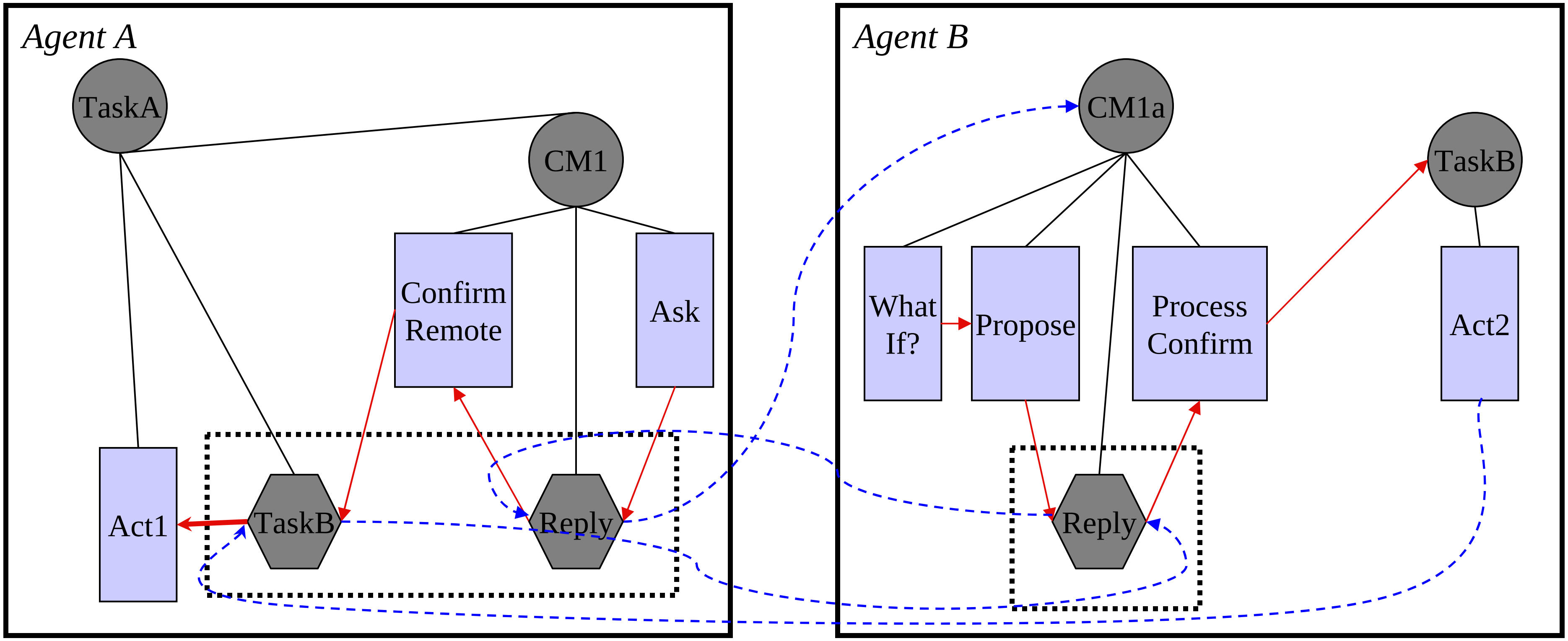


# Example: Coordination by Reservation




What is Act1's Quality, Cost, Duration?
Does Agent B even know I need Act2?

# Example: Coordination by Reservation




*1. When can you finish TaskB? [GPGP Reservation CM Protocol]*

*2. Commit TaskB finish at time t1, quality 34, cost 6.*

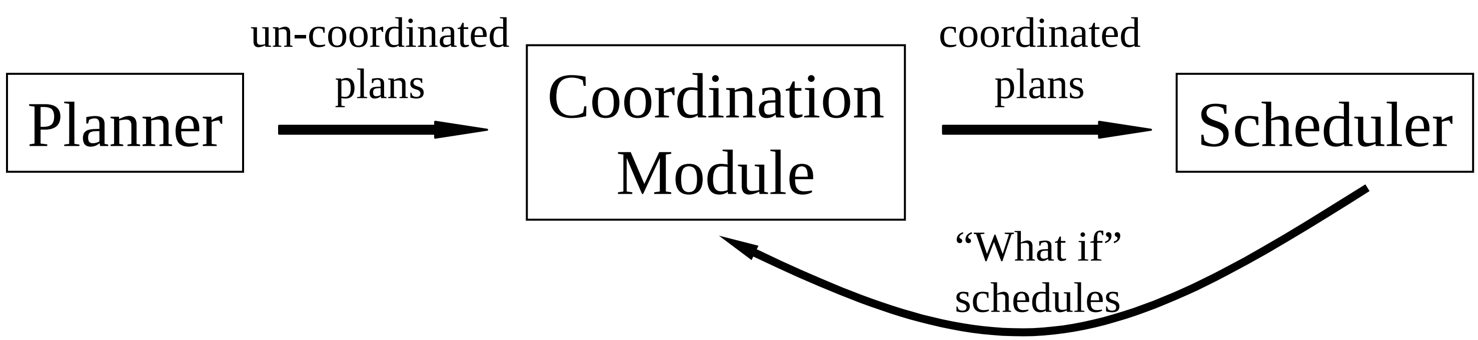
*3. Agreed.*

*4. Here is TaskB's result.*

# Implementation

- Assume agent has local scheduling capability
  - Attempt to maximize utility (self, shared, whatever) by future action sequence
  - Problem is non-local effects make schedule more uncertain or simply unknown (I can't start my task until Agent B does Task B)
- Other assumptions needed for full range of mechanisms
  - Some way to do "what-if" schedule reasoning
  - Ability to make commitments to do, don't, and do w.r.t earliest start times and deadlines
  - Ability to move code for action promotion/demotion

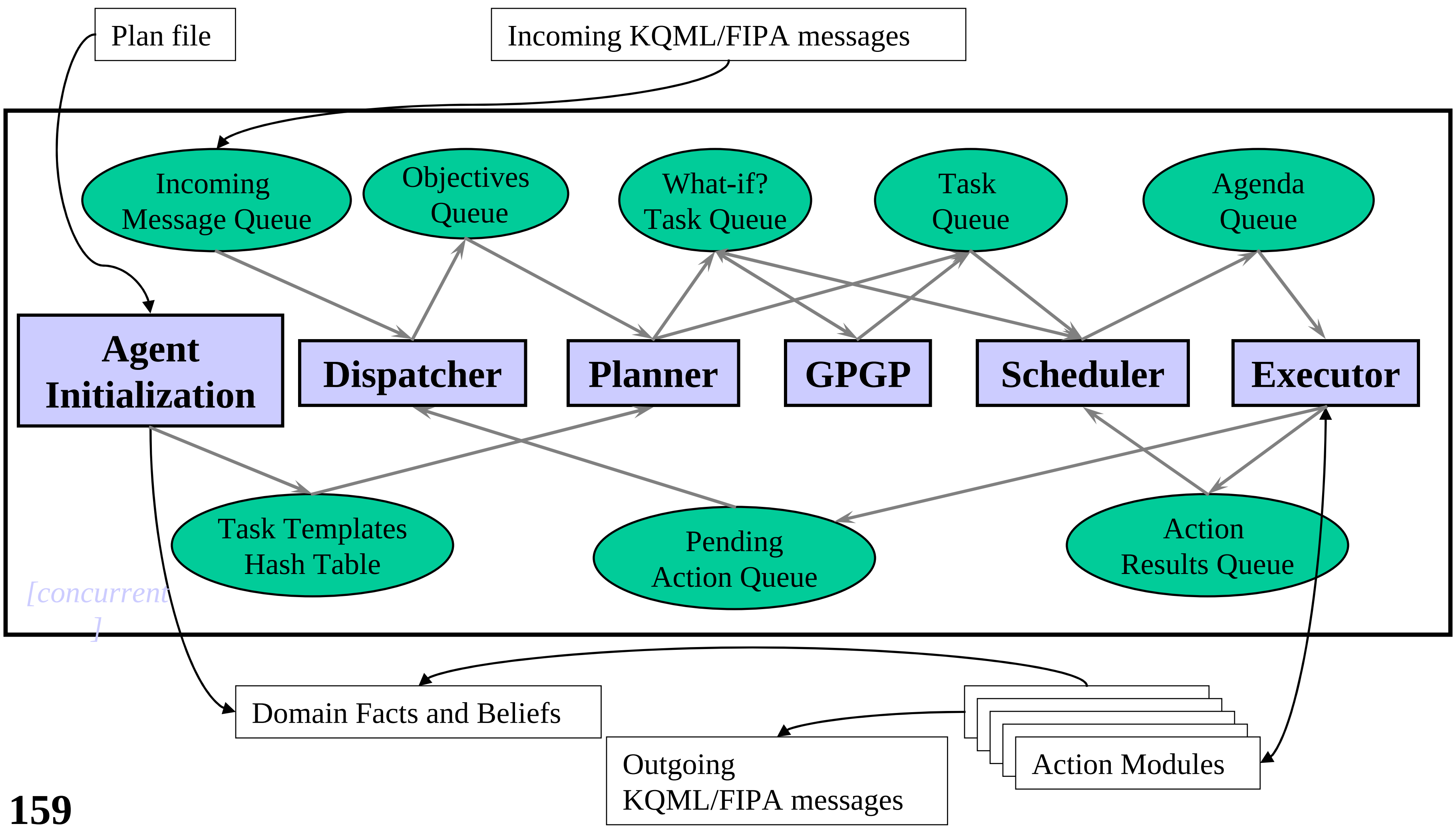
# Coordination Module




**Coordination Module takes advantage of the local scheduler's scheduling ability to evaluate/estimate the features of actions for the remote agents.**

http://www.cis.udel.edu/~decaf

# DECAF Architecture



# CODA: Coordination of Distributed Activities
(Myers, Jarvis, Lee 2001)

**Extends** capabilities of DSIPE

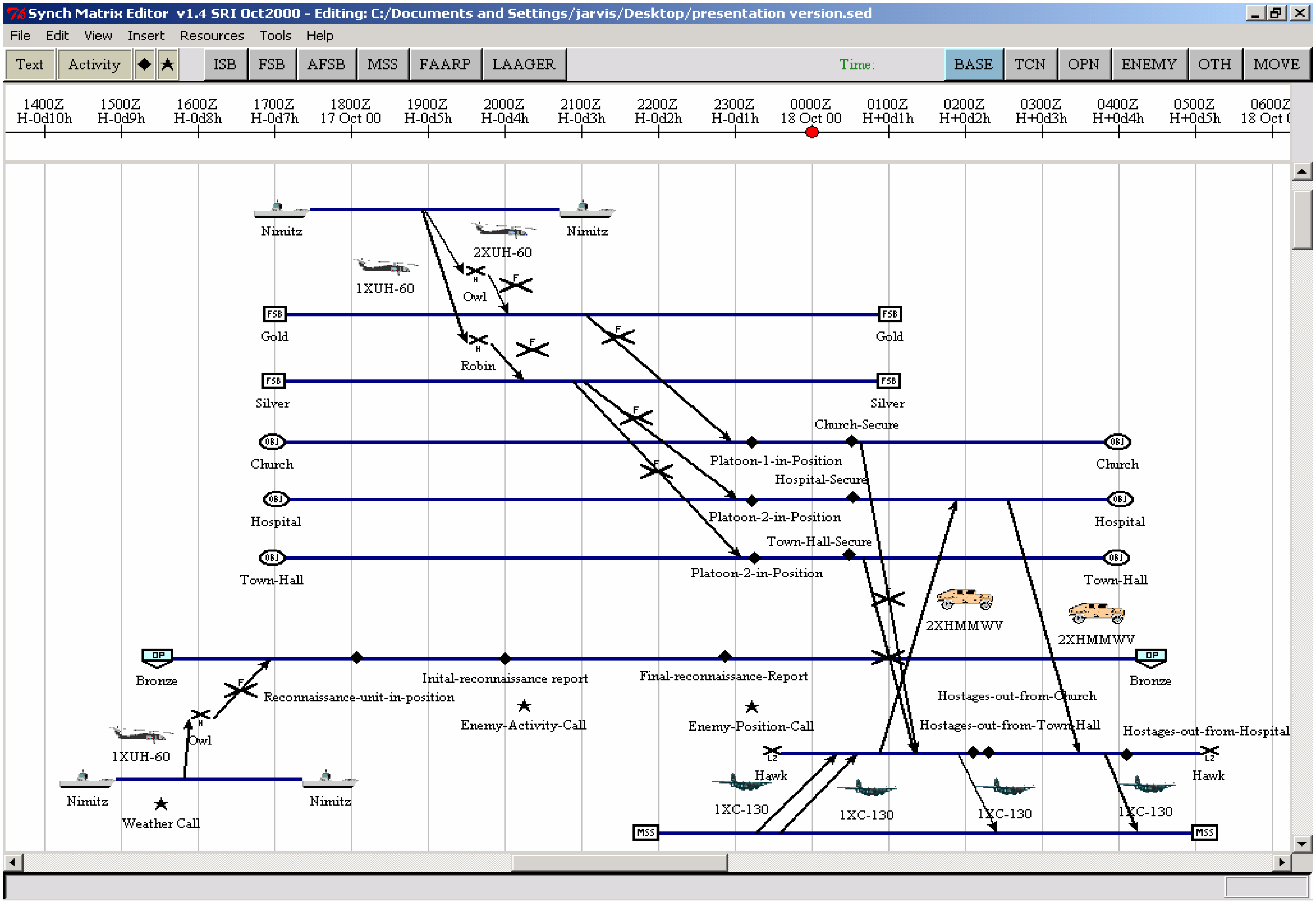
**Objective:** technology for targeted information dissemination

- get the right info to the right people at the right time

**Assumptions**

- Global plan
- Responsibility for subplans delegated to different groups
- Medium-coupling among plans
- Human planning via a *plan-authoring tool*

# SOFTools Plan Authoring System



# Approach

1. Planner Declarations of interest in changes
   – *Plan Awareness Requirements (PARs)*
      - *"Changes in deployment times for transport helicopters"*
      - *"Delay of > 1 hour in evacuating the church"*
2. Unobtrusive Monitoring of Plan Edits
3. Matching of edits to PARs
4. Send notifications

# PAR Types

- CREATION of objects with certain properties
  - *Addition of contingencies related to Weather*

- DELETION of objects with certain properties
  - *Elimination of the Drop Zone south of the Embassy*

- MODIFICATION:

  OBJECTS:
  - *Changes in the use of transport helicopters*

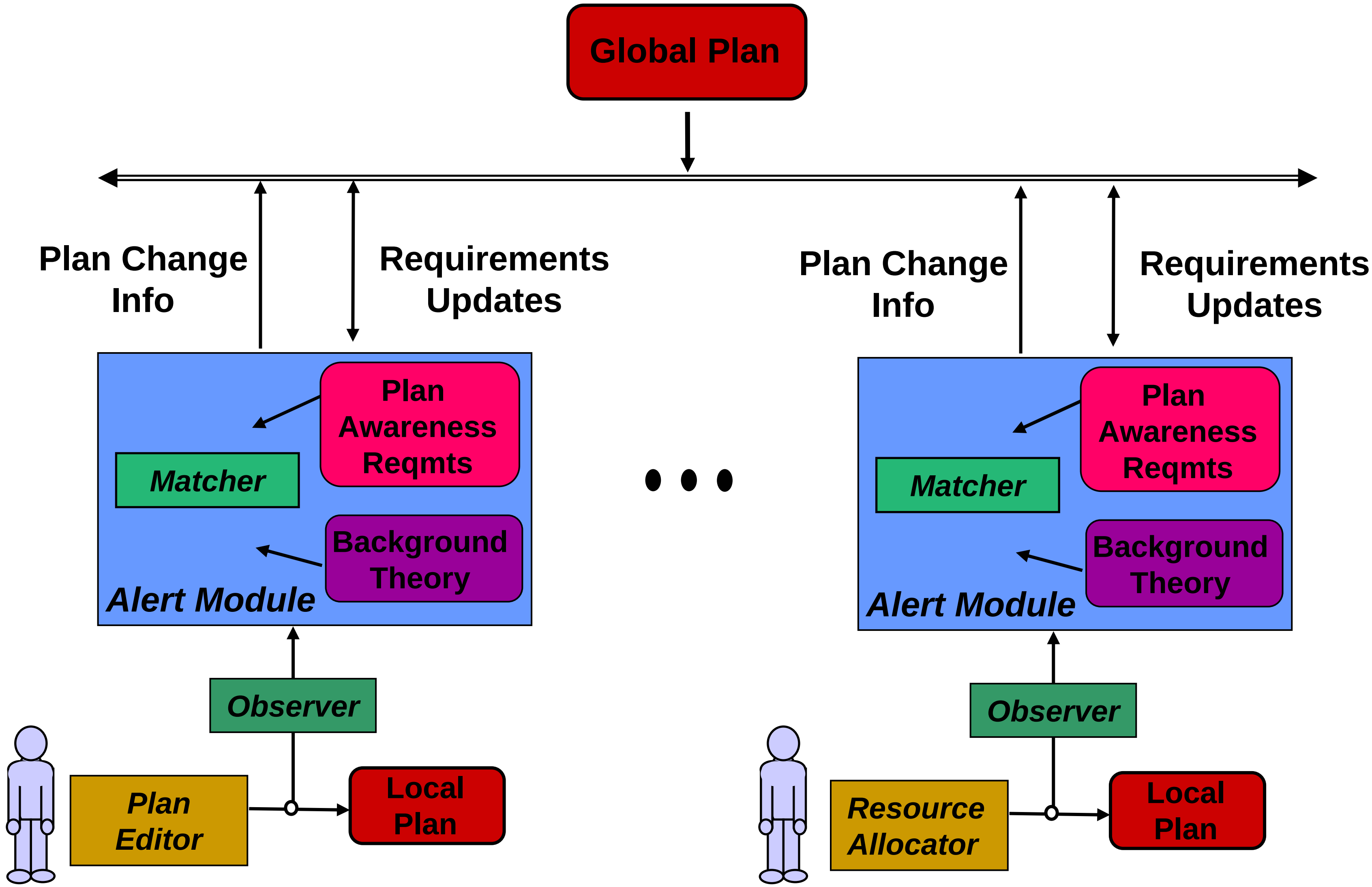
  ATTRIBUTE:
  - *Delay of > 1 hour in expected time to secure Church*

  AGGREGATE:
  - *Decrease of 2 or more in number of UH-60s used*

# Modes of Usage

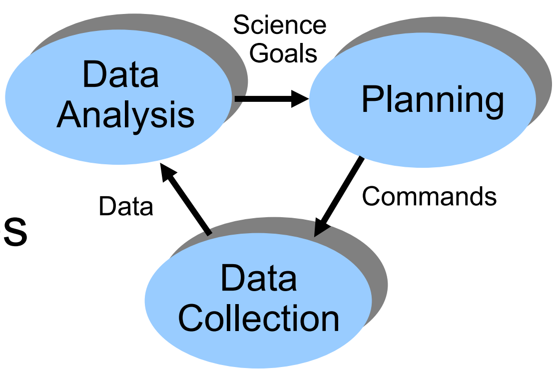
- **Realtime**
  - Instantaneous notification of PAR matches
  - Suitable for:
    - End-phase of planning
    - Execution time
- **Batch Process at Publication Time**
  - Process batches of changes when new versions of plan are published
  - Suitable for:
    - Early- and mid-phase planning

# CODA Architecture



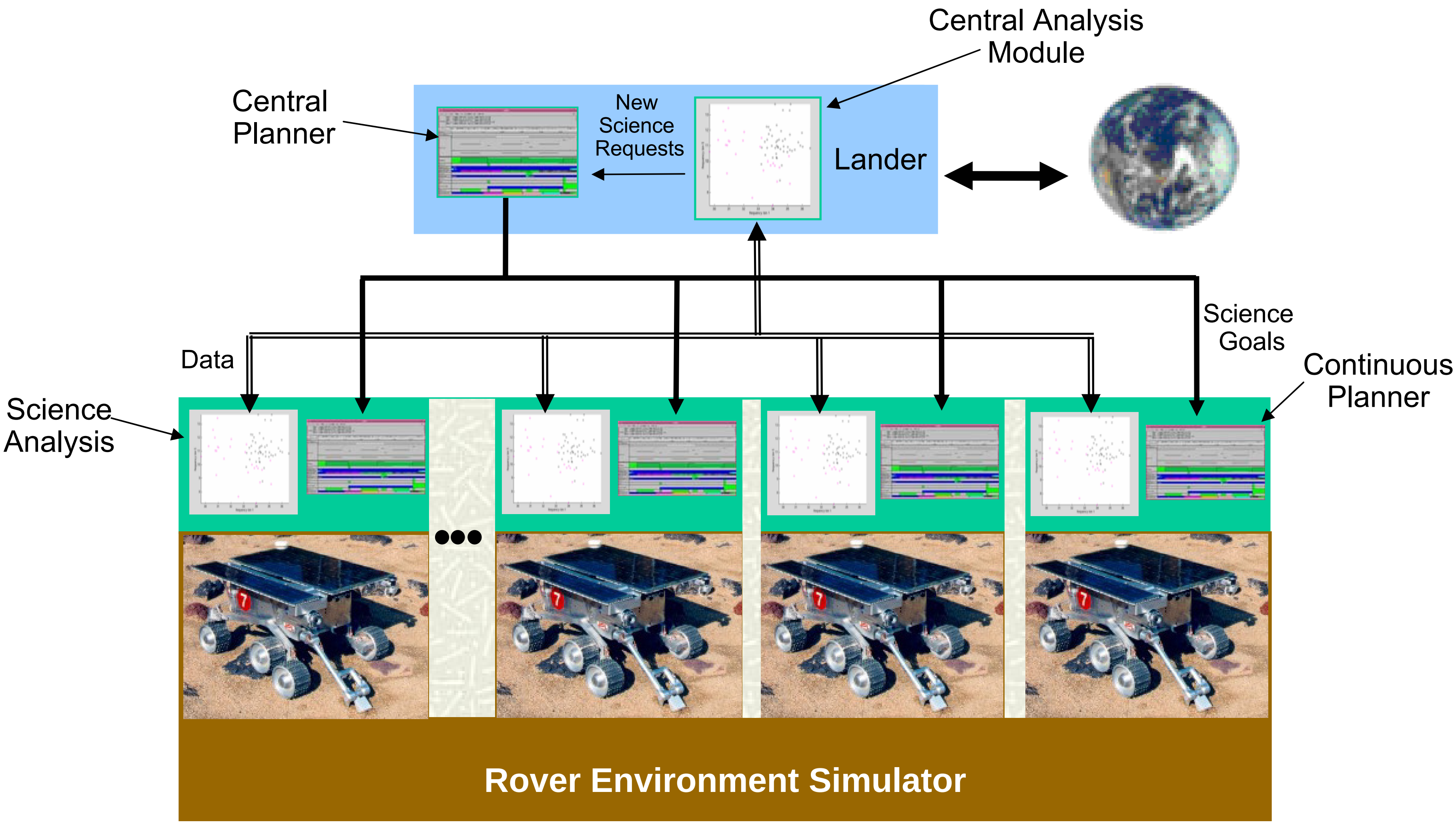
# MISUS
## (Estlin *et al.*, 2000)

- Provides framework for autonomous multi-rover science operations
- System integrates techniques from machine learning and planning/scheduling
  - Data analysis
  - Generation of new science goals and priorities
  - Production of new plans to achieve goals




- System operates in closed-loop fashion to perform science survey with little or no required comm
- Integrated with simulation environment that models planetary terrains

# MISUS Architecture



# Prototype: Science Scenario

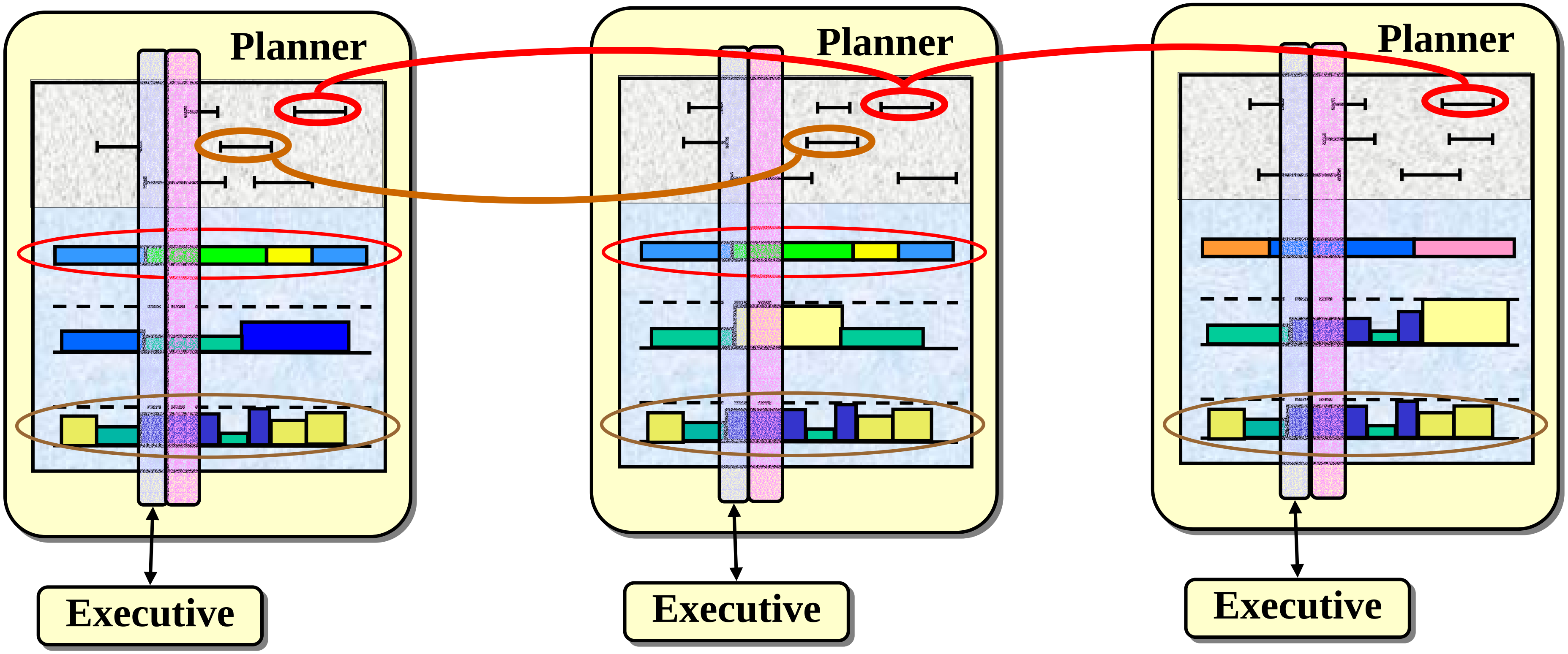
- Goal is to take rock survey of surrounding area
  - Build model of terrain rock distribution
  - Take adequate readings of all rock types
- Three identical rovers
  - Resources: Spectrometer, camera, mast, solar panel, battery
  - Science activities: panoramic spec and camera images, close-up spec and camera images
  - Software: data analysis, planner, control, path planning
- Science readings return set of spectral wavelength values or image texture values

# Shared Activity Coordination

(SHAC, Clement & Barrett, 2003)

– distributed continual planning algorithm
– framework for defining and implementing automated interactions between planning agents (a.k.a. coordination protocols/algorithms)
– software
  - planner-independent interface
  - protocol class hierarchy
  - testbed for evaluating protocols

# Shared Activity Coordination




Shared activities implement team plans, joint actions, and shared states/resources

# Shared Activity Model

- **parameters** (string, integer, *etc*.)
  - **constraints** (e.g. agent4 allows start_time [0,20], [40,50])
- **decompositions** (shared subplans)
- **permissions** - to modify parameters, move, add, delete, choose decomposition, constrain
- **roles** - maps each agent to a local activity
- **protocols** - defined for each role
  - change constraints
  - change permissions
  - change roles
    - includes adding/removing agents assigned to activity

# Delegation Protocol

Delegation::modifyRoles()

– if *roles* does not contain exactly 1 subordinate
  - choose a *subordinate* to whom to delegate the activity
  - add *subordinate* to *roles*

Subordination::modifyRoles()

– if cannot resolve conflicts involving activity
  - remove self from *roles*

# Shared Activity Model

```
shared_activity  mera_communicate
{
  start_time_transmit;
  duration_rcv;
  sender;
  source;
  destination;
  bandwidth;
  size;
  requested_bandwidth;
  bandwidth;
  data_priority;
  requested_delivery_time;
  delivery_time_max;
  delivery_time;
  percent_delivered_overall;
  loss_total_tolerance;
  loss_per_block_tolerance;
  loss_block_size;
  loss_total;
  loss_total_overall;
  loss_per_block;
  loss_per_block_overall;
  prot;

  roles =
    transmit by mera,
    relay by mgs,
    relay by odyssey,
    relay by mex;
```

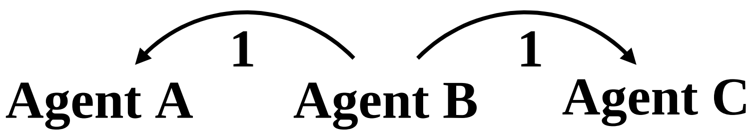
```
protocols =
    mera NetworkDelegation,
    mgs Subordination,
    odyssey Subordination,
    mex Subordination;

permissions =
  mera (all),
  mgs (place, detail, lift, abstract, duration,
       connect, disconnect, parameters),
  odyssey (place, detail, lift, abstract, duration,
           connect, disconnect, parameters),
  mex (place, detail, lift, abstract, duration,
       connect, disconnect, parameters);
};
. . .  // other similar comm activities between
          other spacecraft ommitted

agent mera {
  planner = AspenPlannerInterface(20, 10, 100.0);
  communication = SocketCommunication("ports.txt");
  communicator = AspenCommunicator(comm_windows,
                 comm_window_timeline);
};
. . .  // other similar agent definitions omitted

protocol NetworkDelegation();
protocol Subordination();
```

# SHAC Algorithm

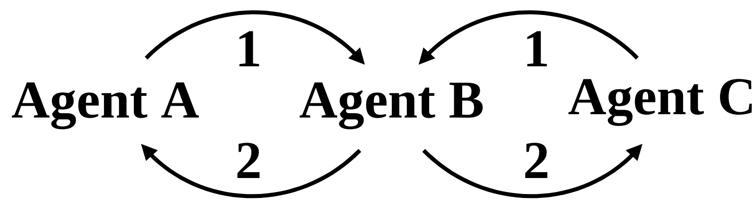
Given: a ***plan*** with multiple activities, including a set of ***shared_activities***, and a ***projection*** of ***plan*** into the future.

1. Revise ***projection*** using the currently perceived state and any newly added goal activities.
2. Alter ***plan*** and ***projection*** while honoring ***constraints*** and ***permissions of shared_activities***.
3. Release relevant near-term activities of ***plan*** to the real-time execution system.
4. For each shared activity in ***shared_activities***
   – apply each associated *protocol* to modify the activity
5. Communicate changes in ***shared_activities***.
6. Update ***shared_activities*** based on received communications.
7. Go to 1.

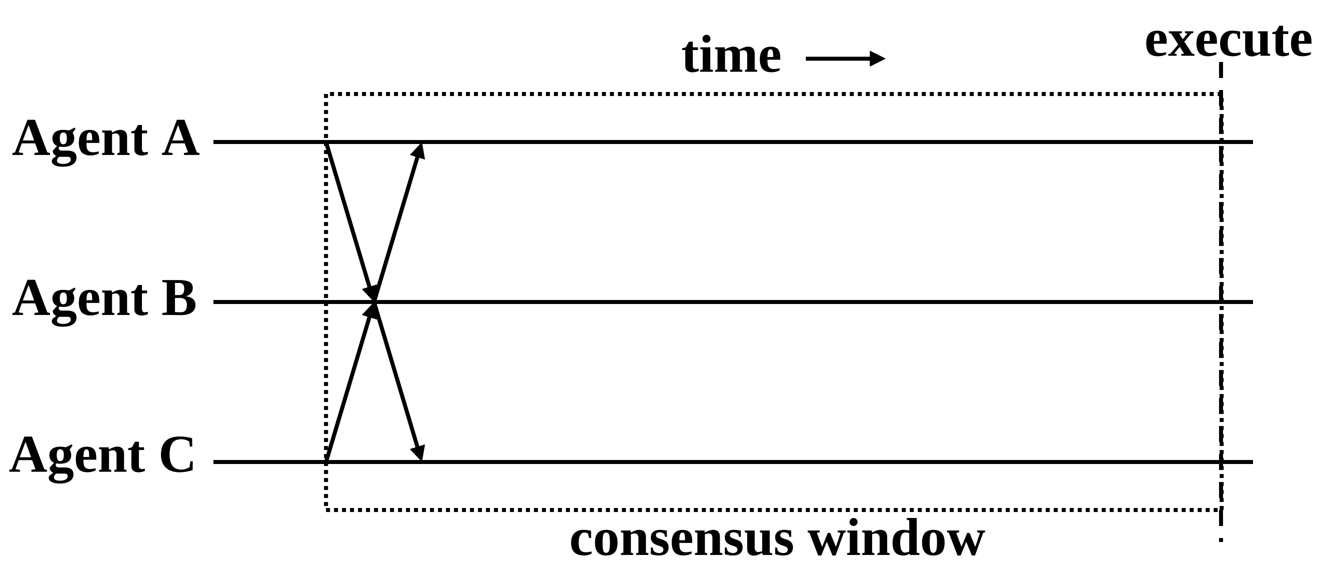
# Computing Consensus Windows



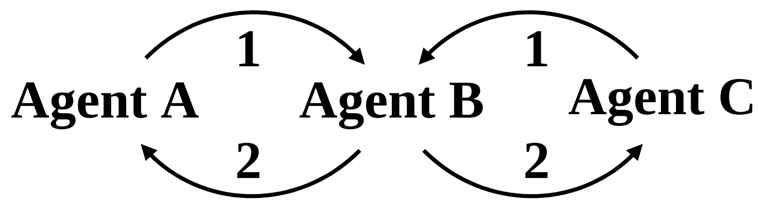

highest rank decides



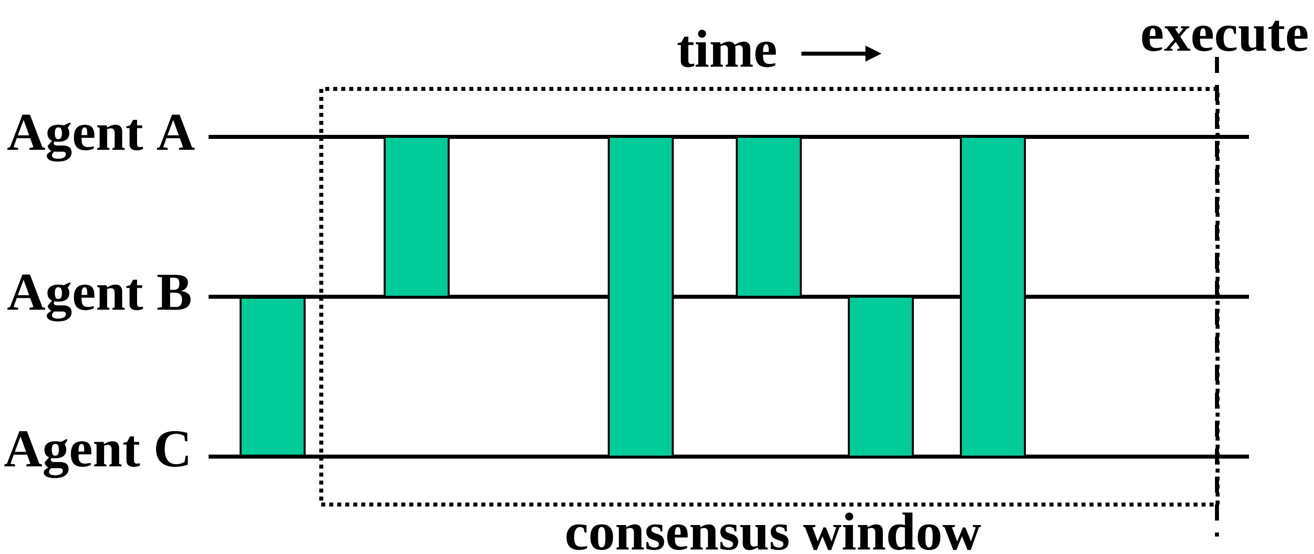

voting or auction



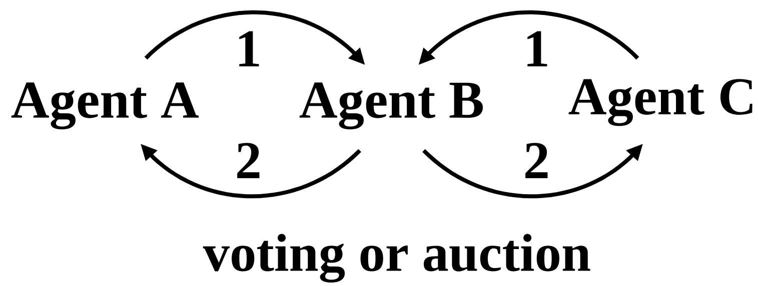
# Computing Consensus Windows

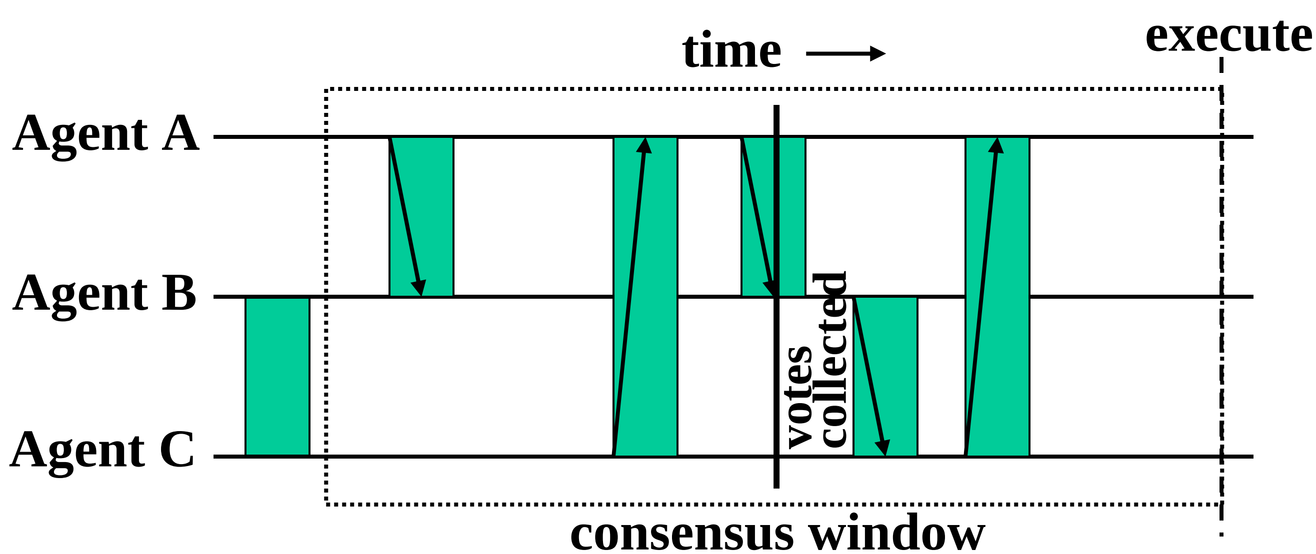



voting or auction

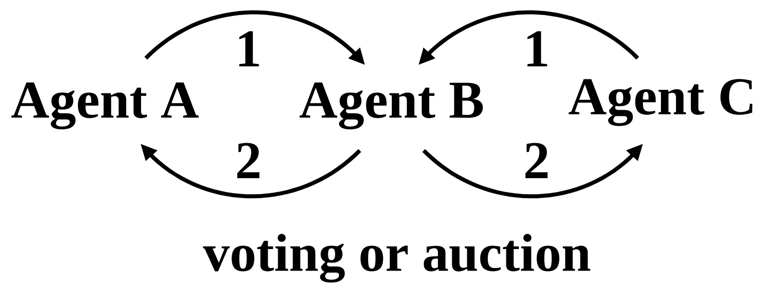


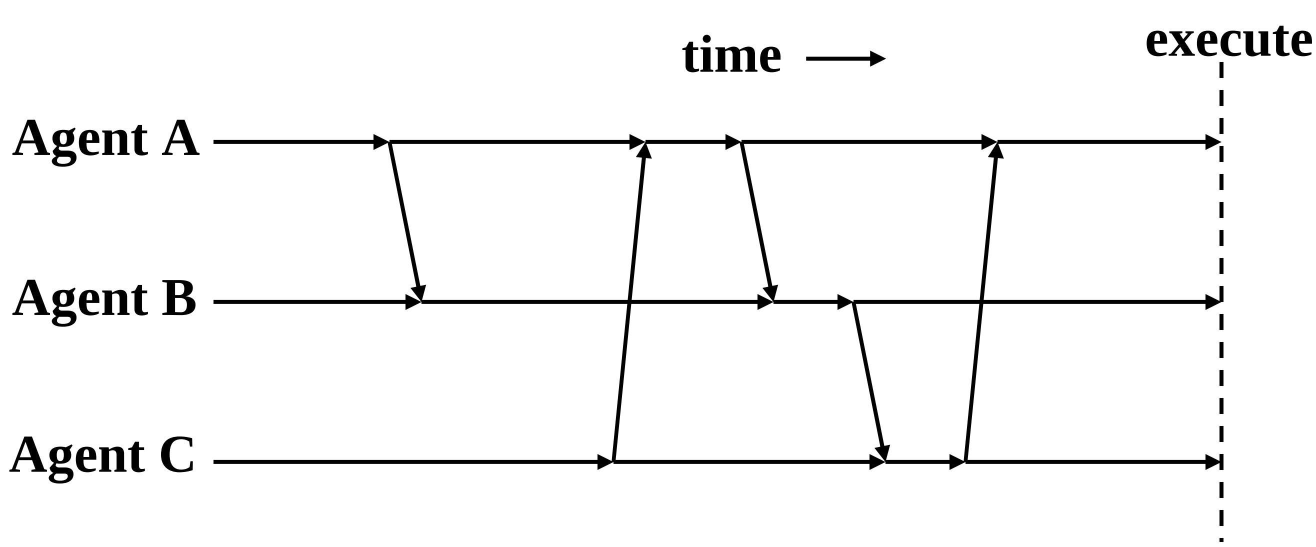
# Computing Consensus Windows



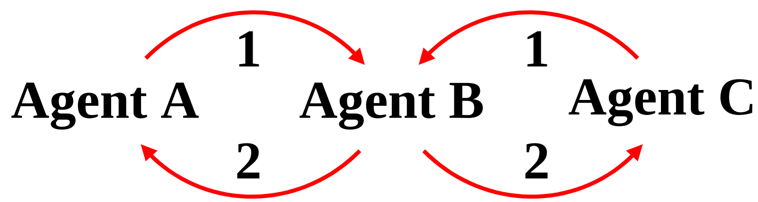



# Computing Consensus Windows

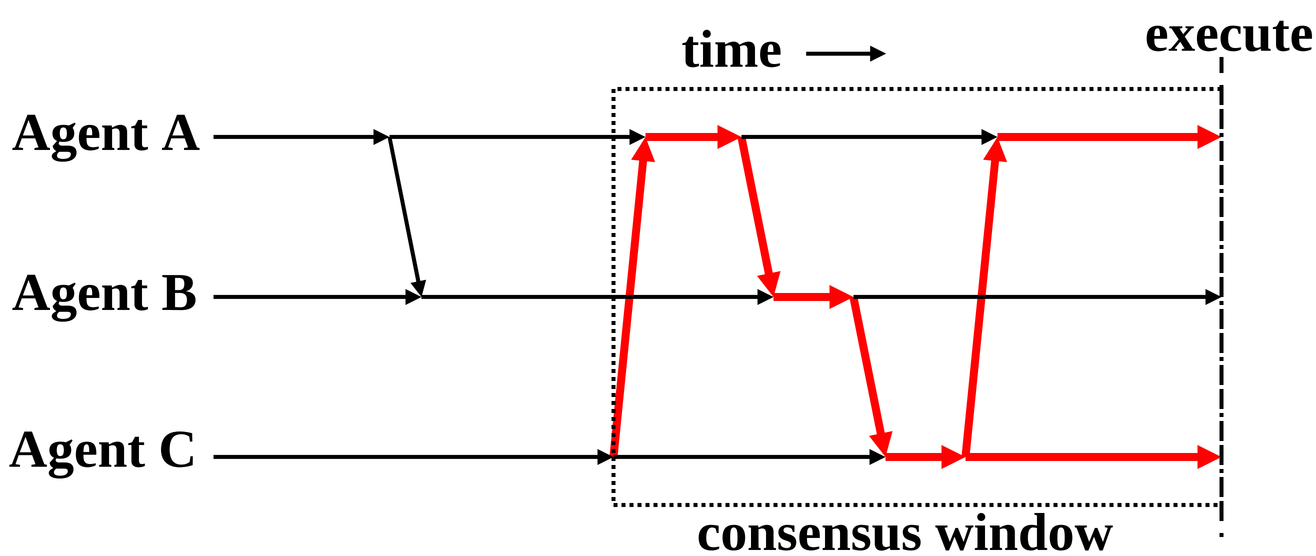


# Computing Consensus Windows

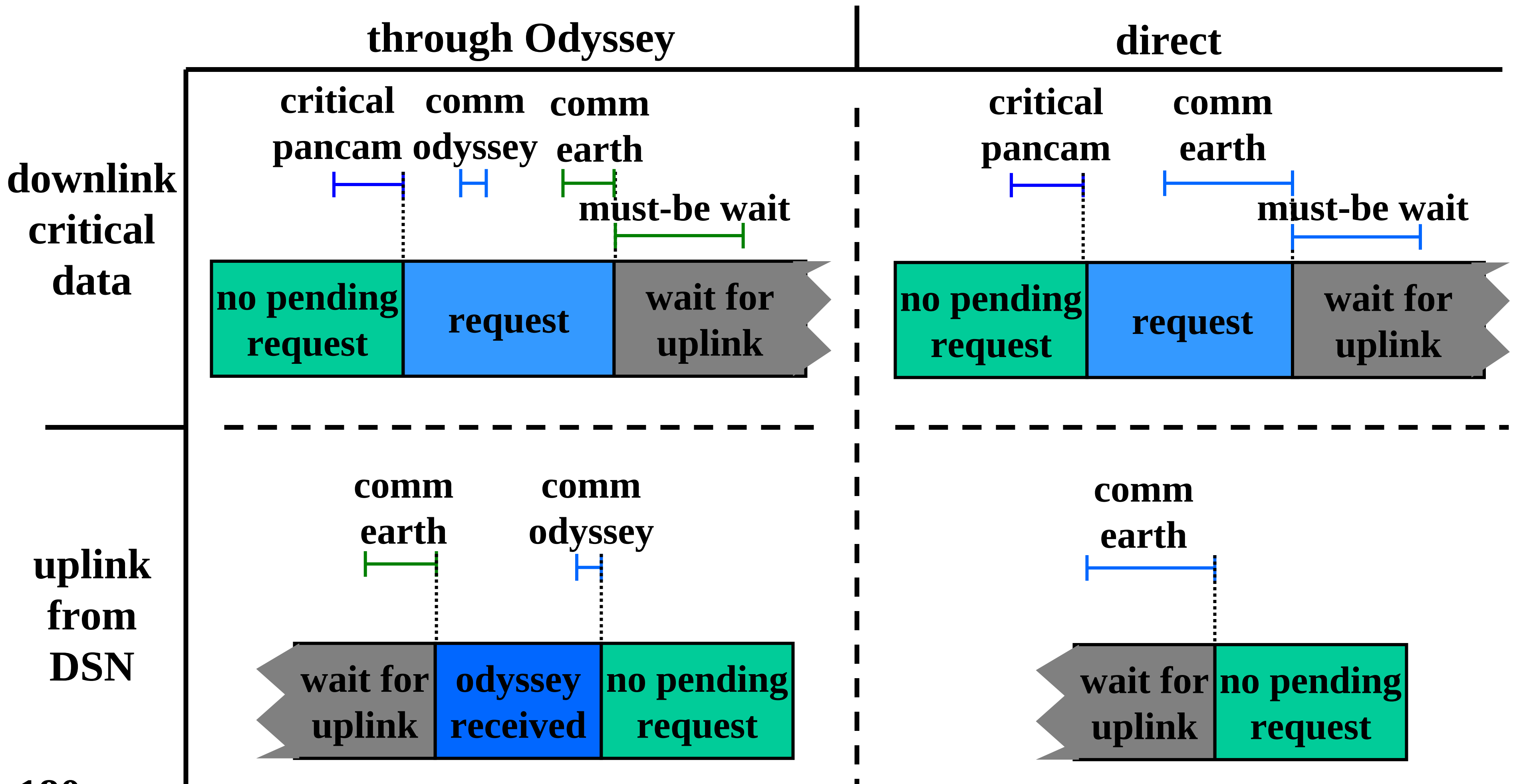



voting or auction

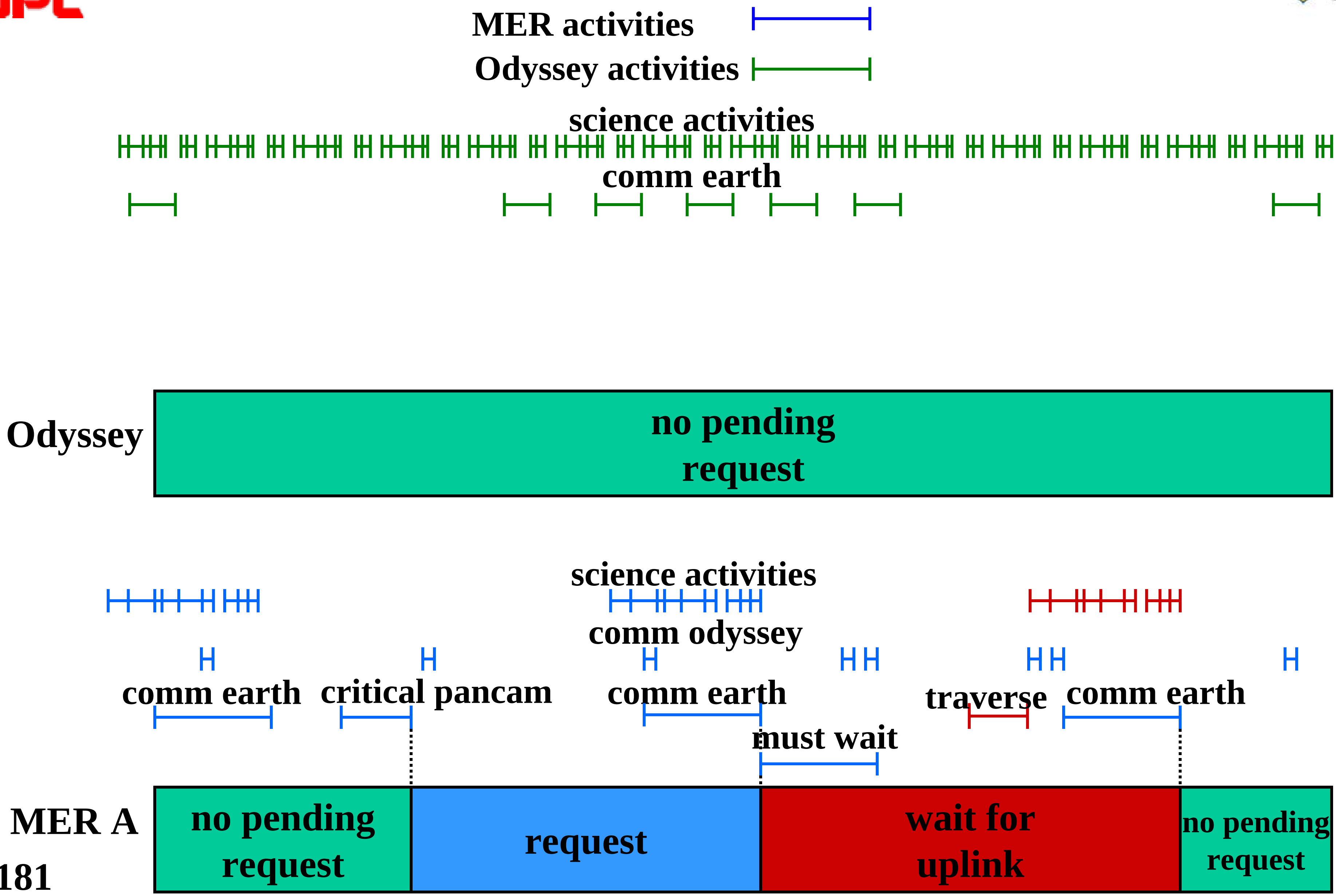


# Mars Scenario

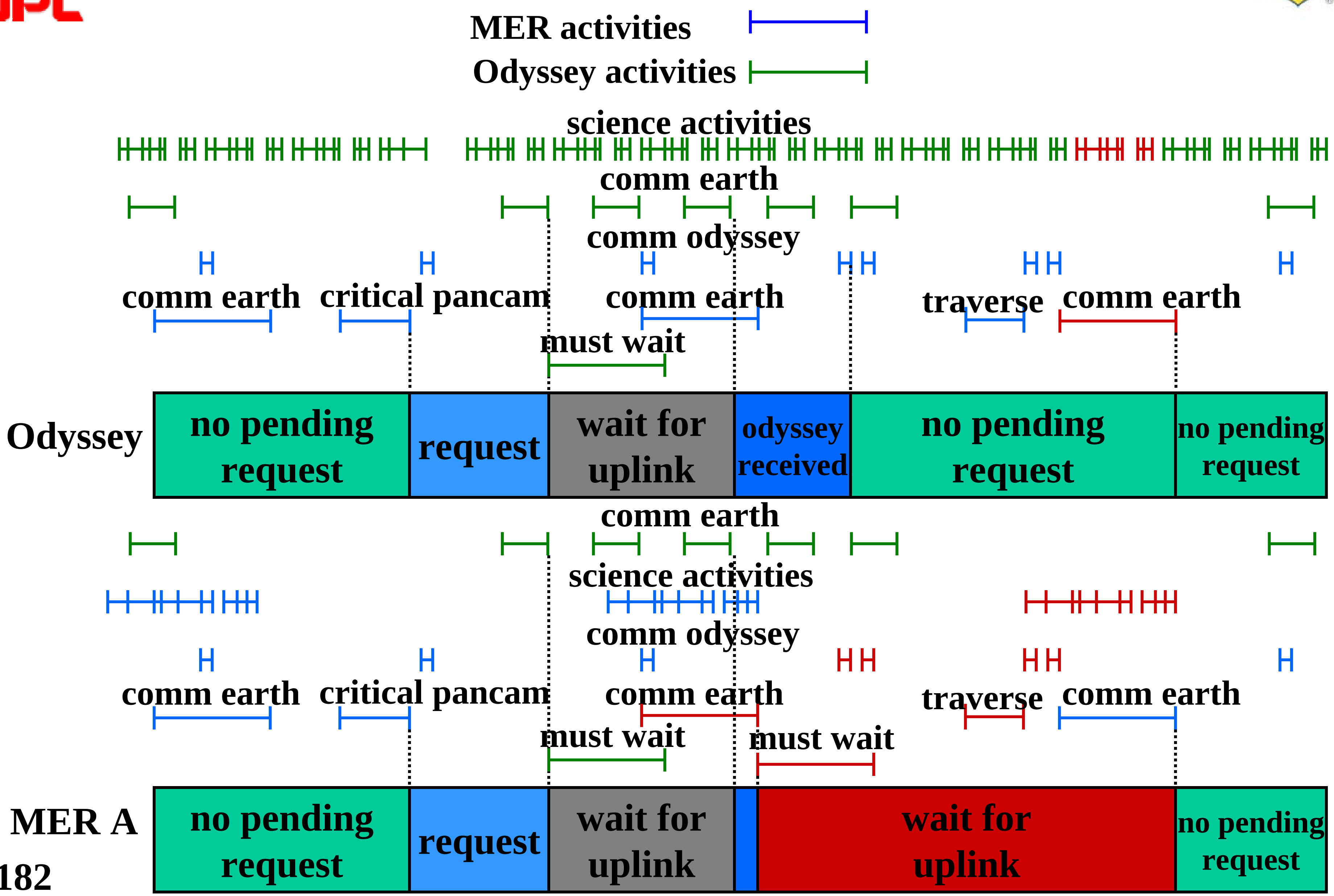
MER activities
Odyssey activities



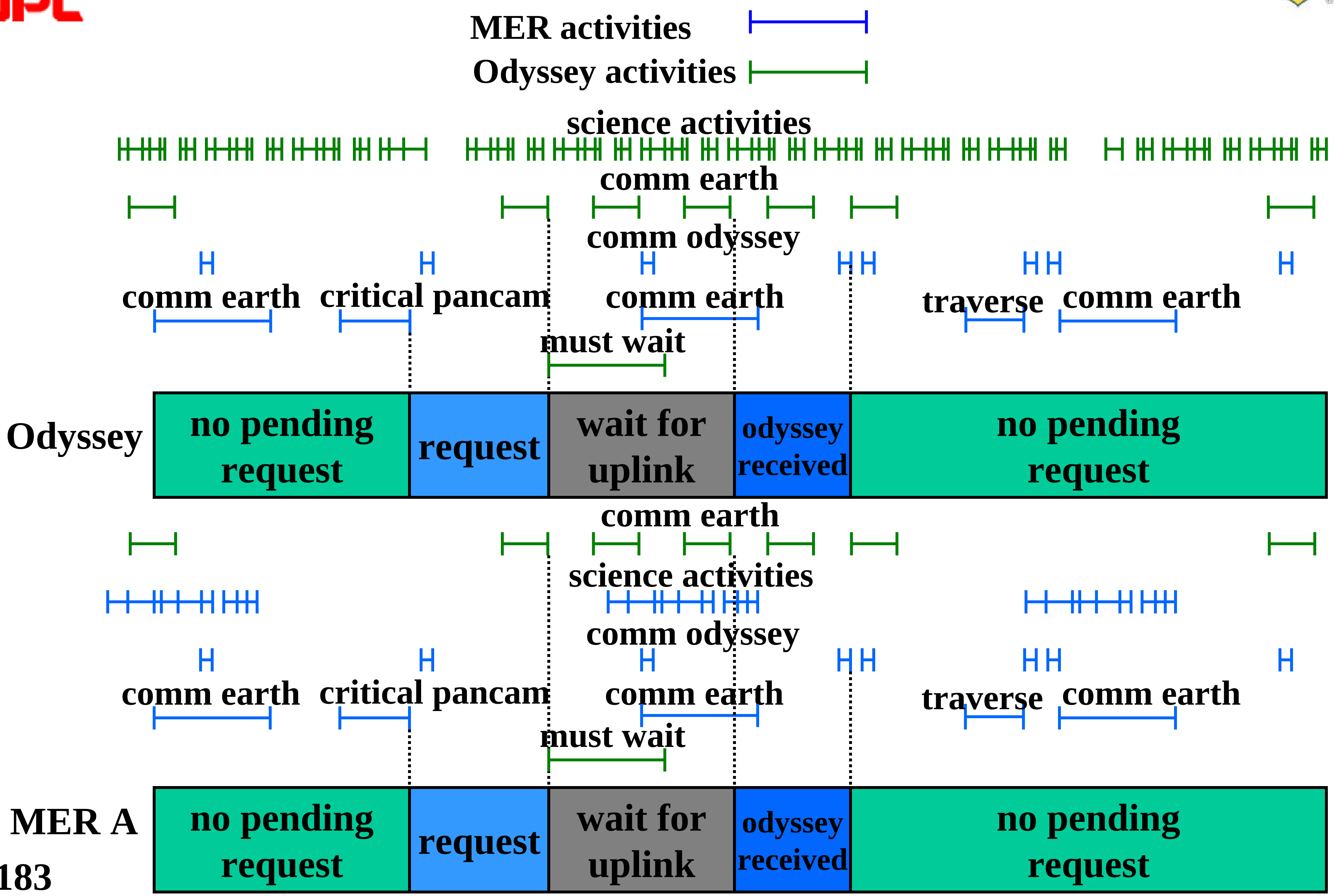
# Mars Scenario



# Mars Scenario



# Mars Scenario



# Protocol Capabilities

Defining/extending protocol classes

1. modify permissions
2. modify local parameter constraints
3. add/delete sharing agents
4. change roles of sharing agents

Default protocol class

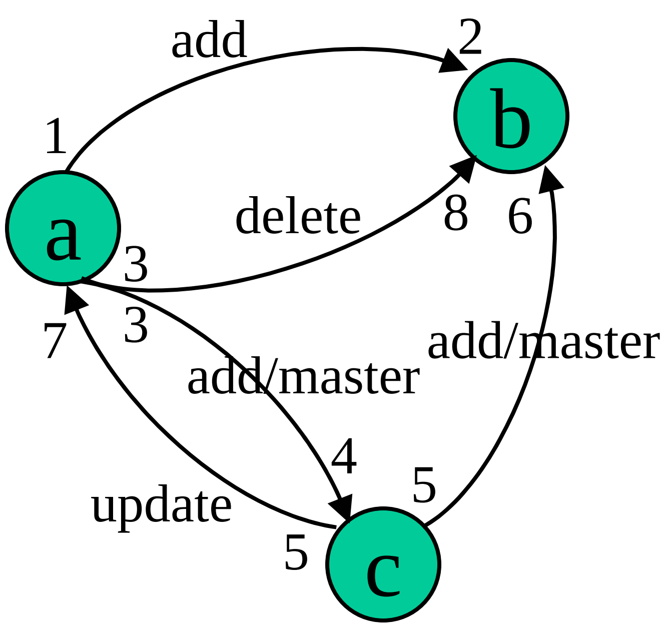
- joint intention
- mutual belief
- resource sharing
- active/passive roles
- master/slave roles

# Control Protocols for a Shared Activity

- Chaos
  - A free-for-all among planners
- Master/Slave
  - The master has permissions, slaves don't
- Round Robin
  - Master role passes round-robin among planners
- Asynchronous Weak Commitment (AWC)
  - Neediest planner becomes master

- Variations
  - how many planners share activity
  - use of constraints

# Causal Inconsistency




Order of events

1. a is master and shares with (adds to roles) b
2. b receives add from a
3. a replaces b with c and makes c master
4. c receives add message making it master
5. c makes b master and removes self (deletes)
6. b receives add/master from c (before delete from a)
7. a receives update from c
8. b receives delete from a

A SHAC protocol is proven sound if

- the underlying planners are sound,
- the protocol ensures that only one agent has permissions over any piece of information, and
- it employs causally consistent communication

# References


- W. Briggs and D. J. Cook, Flexible Social Laws, *Proceedings of the International Joint Conference on Artificial Intelligence*, 1995
- S. Cammarata, D. McArthur, and R. Steeb, “Strategies of cooperation in distributed problem solving,” in Proceedings of the Eighth International Joint Conference on Artificial Intelligence, (Karlsruhe, Germany), pp. 767–770, Aug. 1983.
- Clement, B. and Barrett, A. Continual Coordination through Shared Activities. *Proceedings of the Second International Conference on Autonomous Agents and Multi-Agent Systems*, 2003.
- Clement, B. and Durfee, E. Theory for Coordinating Concurrent Hierarchical Planning Agents Using Summary Information. *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, pp. 495-502, 1999.
- P. R. Cohen and H. J. Levesque, “Intention is choice with commitment,” Artificial Intelligence, vol. 42, no. 3, pp. 213–261, 1990.
- Corkill, D. Hierarchical Planning in a Distributed Environment. *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, pp. 168-175, Tokyo, August 1979.
- K. Decker, S. Khan, C. Schmidt, G. Situ, R. Makkena, and D. Michaud, “Biomas: A multi-agent system for genomic annotation,” International Journal of Cooperative Information Systems, vol. 11, no. 3–4, 2002.
- K. Decker and J. Li, “Coordinating mutually exclusive resources using GPGP,” Autonomous Agents and Multi-Agent Systems, vol. 3, no. 2, pp. 133–157, 2000.
- Decker K. and Lesser, V. Designing a Family of Coordination Algorithms. In *Proceedings of the First International Conference on Multi-Agent Systems*, San Francisco, July 1995.

# References


- desJardins, M. and Wolverton, M. Coordinating a distributed planning system. *AI Magazine*, 20(4):45–53, 1999.
- Durfee, E. and Lesser, V. Partial Global Planning: A Coordination Framework for Distributed Hypothesis Formation. *IEEE Transactions on Systems, Man, and Cybernetics, Special Issue on Distributed Sensor Networks*, SMC-21(5):1167-1183, September 1991.
- E. Ephrati and J. Rosenschein, “Divide and conquer in multi-agent planning,” in Proceedings of the Twelfth National Conference on Artificial Intelligence, (Seattle), pp. 375–380, AAAI Press/MIT Press, 1994.
- M. Georgeff, “A theory of action for multiagent planning,” in Proceedings of the Fourth National Conference on Artificial Intelligence, pp. 121–125, Aug. 1984.
- M. Georgeff, “Communication and interaction in multiagent planning,” in Proceedings of the Eighth International Joint Conference on Artificial Intelligence, pp. 125–129, Aug. 1983.
- C.A. Goble, S. Pettifer, R. Stevens and C. Greenhalgh *Knowledge Integration: In silico Experiments in Bioinformatics* in The Grid: Blueprint for a New Computing Infrastructure Second Edition eds. Ian Foster and Carl Kesselman, 2003, Morgan Kaufman, November 2003.
- J. Graham, K. Decker, M. Mersic. DECAF: A Flexible Multi-Agent System Architecture. Autonomous Agents and Multi-Agent Systems. 7(1–2): 7–27, 2003.

# References


- Grosz, B. and Kraus, S. Collaborative Plans for Complex Group Action. In *Artificial Intelligence*. 86(2), pp. 269-357, 1996.
- N. R. Jennings. Specification and implementation of belief, desire, *joint-intention architecture for collaborative problem solving*. Journal of Intelligent and Cooperative Information Systems, 2(3):289--318, 1993.
- Lansky, A. Localized Search for Multi-agent Domains, *Proc. 12th Int'l Joint Conf. Artificial Intelligence (IJCAI '91)*, pp. 252-258, 1991.
- V. Lesser, K. Decker, T. Wagner, N. Carver, A. Garvey, B. Horling, D. Neiman, R. Podorozhny, M. NagendraPrasad, A. Raja, R. Vincent, P. Xuan, X.Q. Zhang. Evolution of the GPGP/TÆMS Domain-Independent Coordination Framework. In Autonomous Agents and Multi-Agent Systems. 9 (1-2): 87-143, July - September, 2004T.
- W. Malone and K. Crowston, "Toward an interdisciplinary theory of coordination," Center for Coordination Science Technical Report 120, MIT Sloan School of Management, 1991.
- F. v. Martial, Coordinating Plans of Autonomous Agents. Berlin: Springer-Verlag, Lecture Notes in Artificial Intelligence no. 610. 1992.
- Myers, K., Jarvis, P., and Lee, T. CODA: Coordinating Human Planners. *Proceedings of the European Conference on Planning*, Toledo, Spain, 2001.
- Shoham, Y. and Tennenholtz, M. On the synthesis of useful social laws for artificial societies. *Proceedings of the National Conference on Artificial Intelligence*, pages 276–281, 1992.
- Tambe, M. Toward Flexible Teamwork. *Journal of Artificial Intelligence Research, Volume 7, Pages 83-124,* 1997.