Coordination of Highly Contingent Plans

David Musliner — Honeywell

Ed Durfee, Jianhui Wu, Dmitri Dolgov — Michigan

Robert Goldman — SIFT

Mark Boddy — Adventium Labs

Outline

- Problem overview: Coordinators, C-TAEMS.
  - Relationship to prior talks:
    - Distributed coordination of teams.
    - Dynamic changes, on-the-fly replanning.
    - Things that are connected in plans are Nodes (for Austin!)
- Agent design overview.
- Mapping single-agent task models to MDPs.
- Achieving inter-agent coordination.
- Lessons and future directions.

Motivating Problem

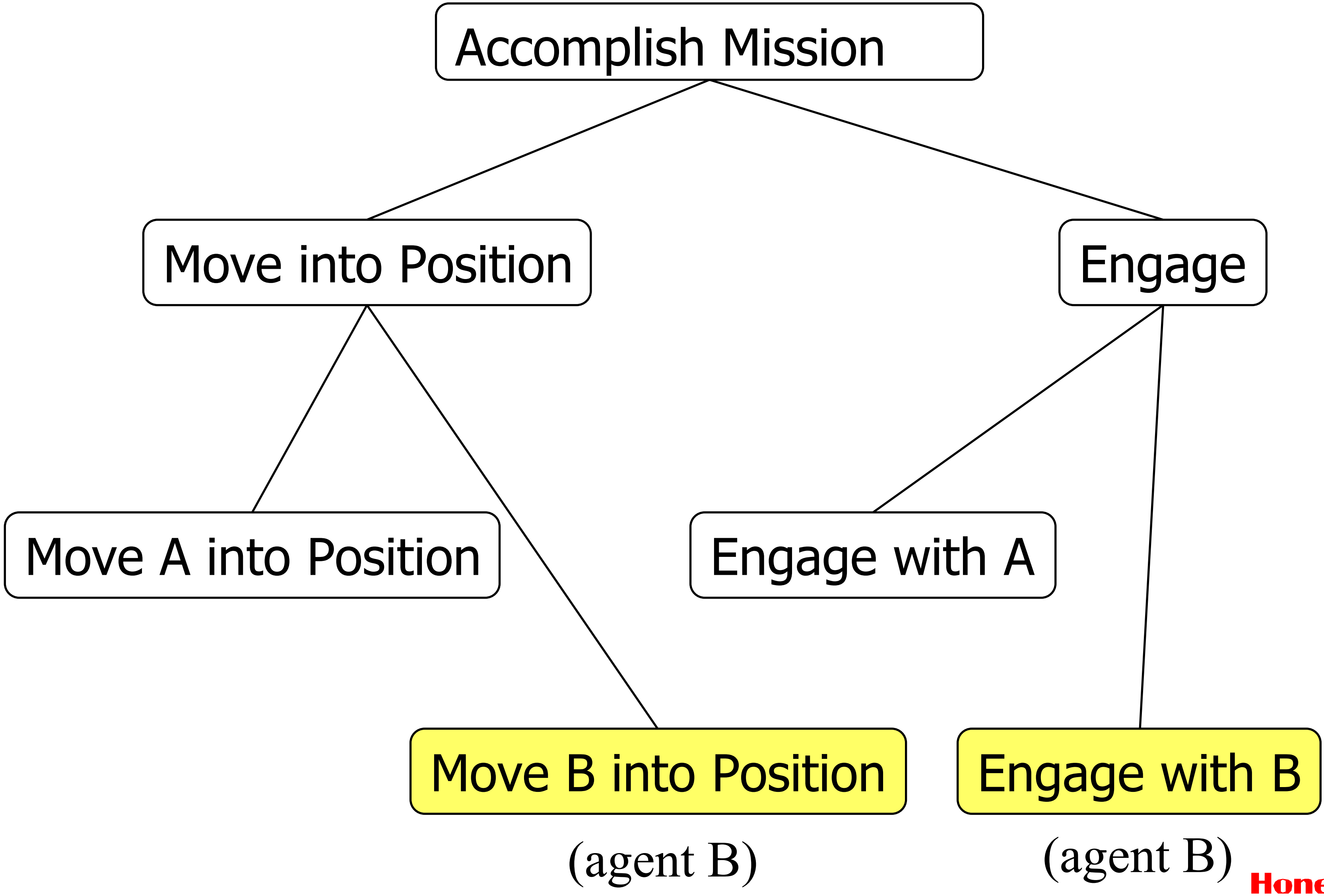
- Coordination of mission-oriented human teams, at various scales.
  - First responders (e.g., firefighters).
  - Soldiers.
- Distributed, multi-player missions.
- Complex interactions between tasks.
- Uncertainty in task models – both duration and outcome.
- Dynamic changes in tasks: unmodeled uncertainty, new tasks.
  → Highly contingent plans = policies.
  → More powerful representation than current military plans.
- Real-time.

- Challenge: the right tasks done by the right people at the right time.

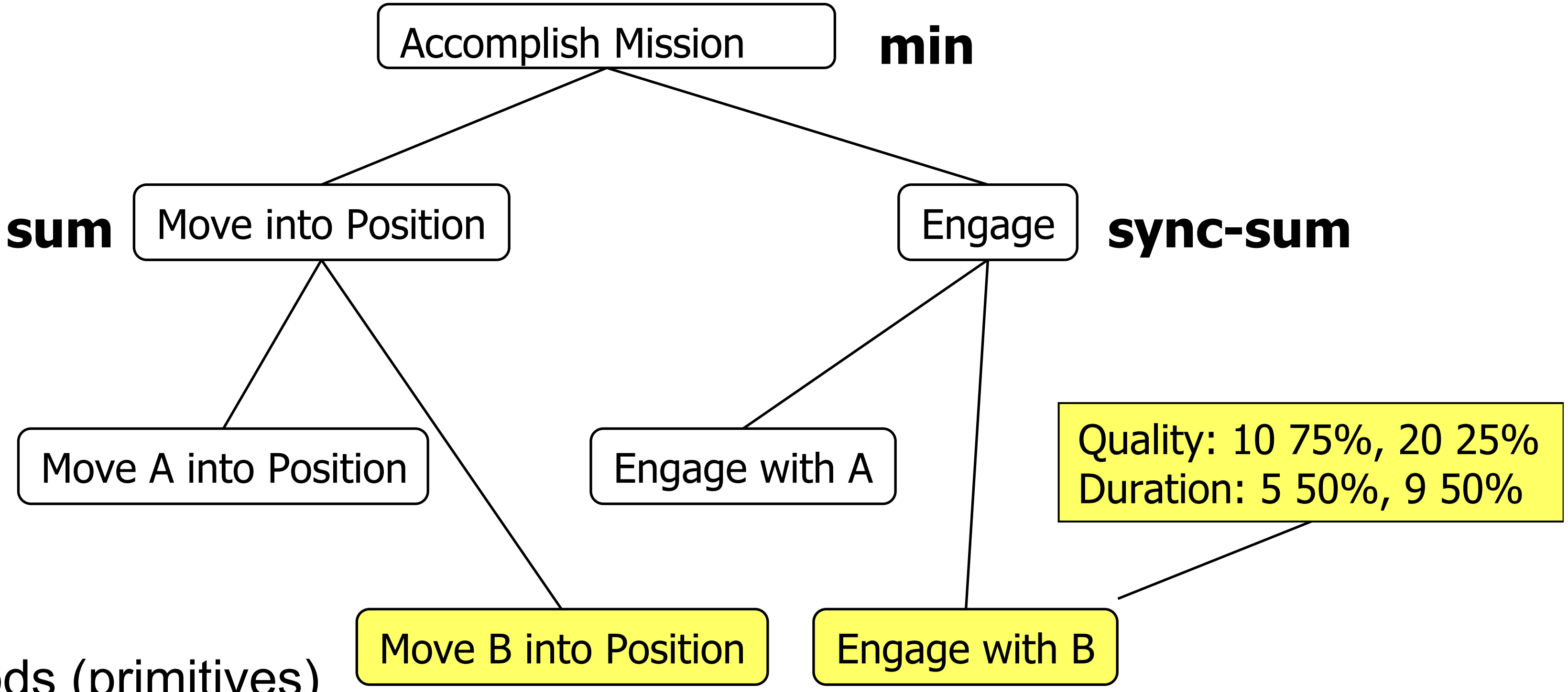
CTAEMS- A Language and Testbed

- CTAEMS is a hierarchical task model used by the Coordinators program.
- Stresses reasoning about the *interactions* between tasks and about the *quality* of solutions.
- There is no explicit representation of world state, unlike conventional plan representations.

CTAEMS Includes Task Decomposition



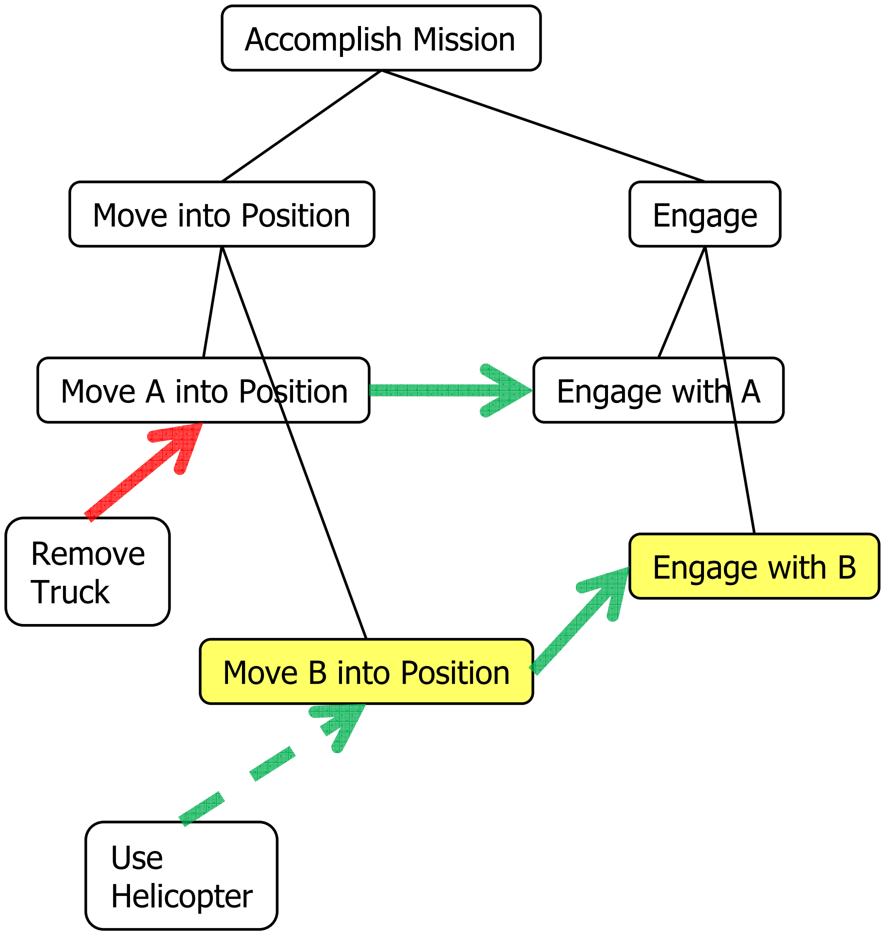
Modeled Uncertainty & Quality Accumulation



- Methods (primitives)
  - are temporally-extended, with deadlines and release times;
  - are stochastic, with multiple outcomes.
  - Each agent can perform only one at a time.
- *Tasks* have *QAFs* used to roll-up quality from children.
- Root node quality is overall utility.

Non-local Effects (NLEs)

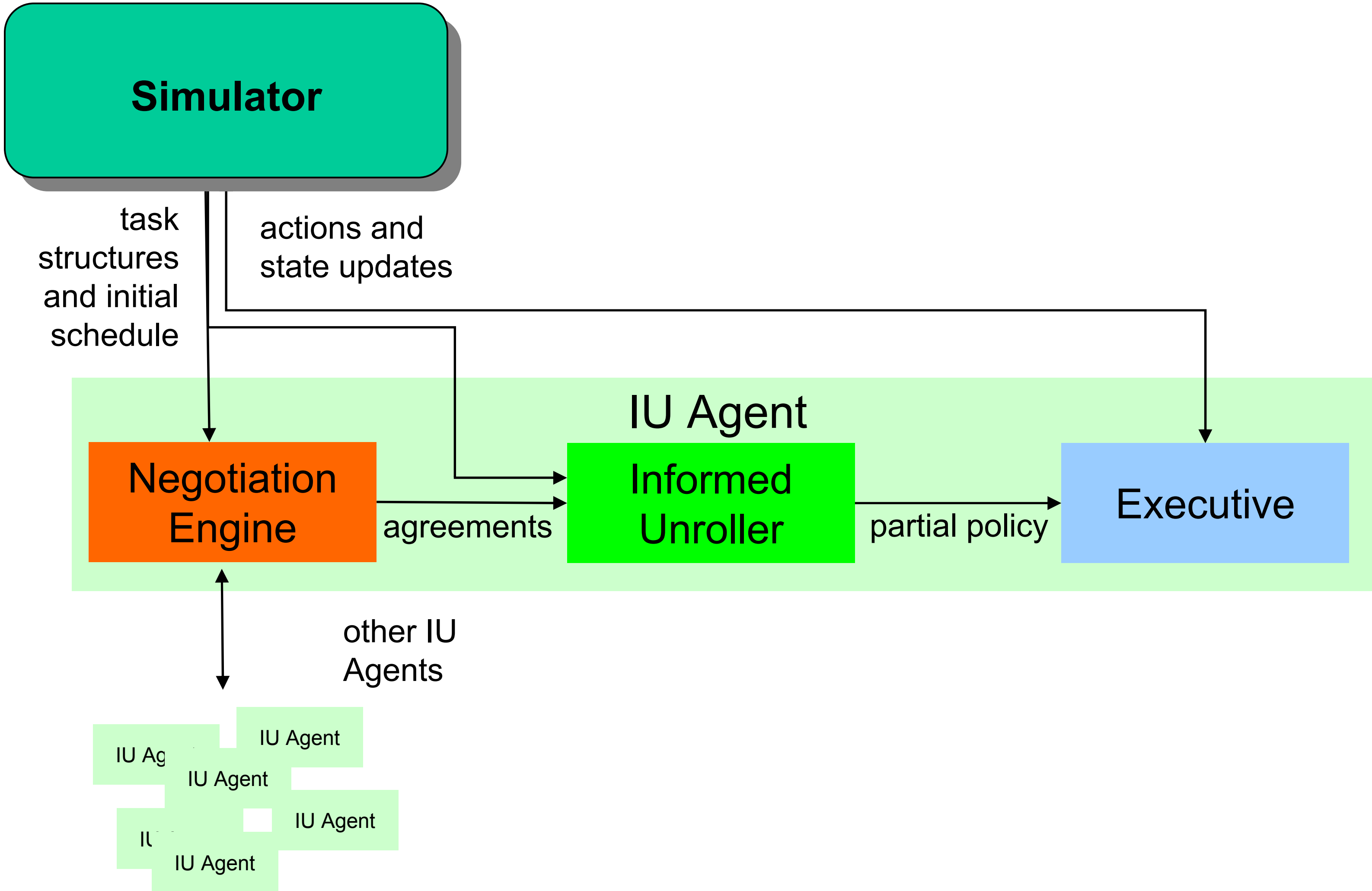
- Non-local effects (NLEs) are edges between nodes in the task net.
- The quality of the source node will affect the target node.
- NLEs can be positive or negative and qualitative or quantitative:
  - Enablement, disablement, facilitation or hindering.
- These effects can also be *delayed*.



Approach Overview

- "Unroll" compact CTAEMS task model into possible futures (states) in a probabilistic state machine – a Markov Decision Process.
- MDPs provide a rigorous foundation for planning that considers uncertainty and quantitative reward (quality).
  - State machines with reward model, uncertain actions.
  - Goal is to maximize expected utility.
  - Solutions are *policies* that assign actions to every reachable state.
- Distribution is fairly new: single-agent MDPs must be adapted to reason about multi-agent coordination.
- Also, CTAEMS domains present the possibility of meta-TAEMS task model changes and un-modeled failures.
- Honeywell's IU-Agent addresses these concerns (partly).

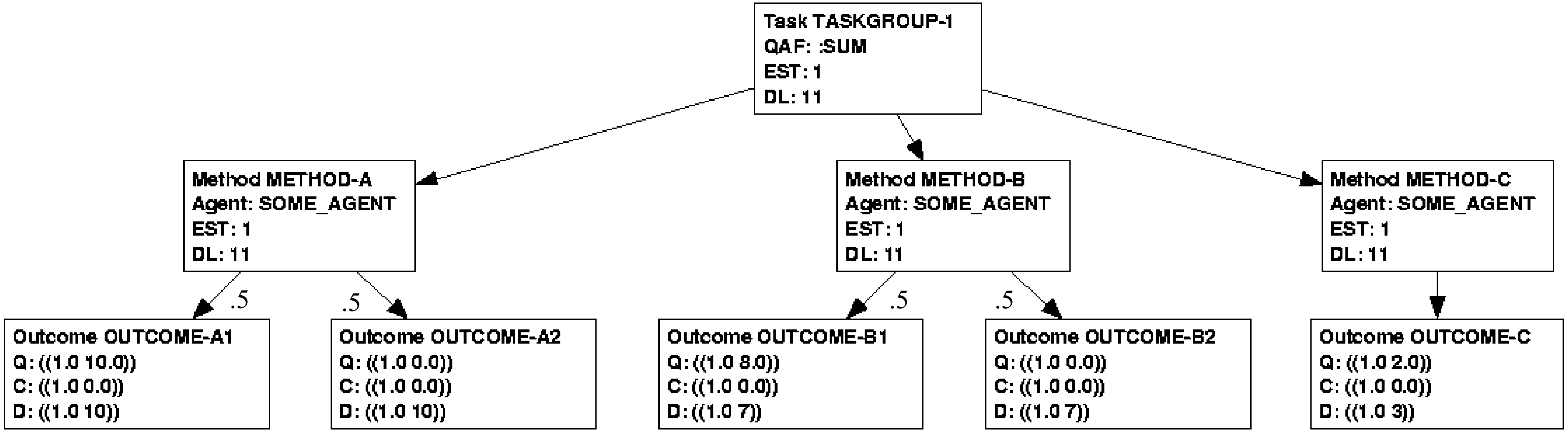
IU Agent Architecture



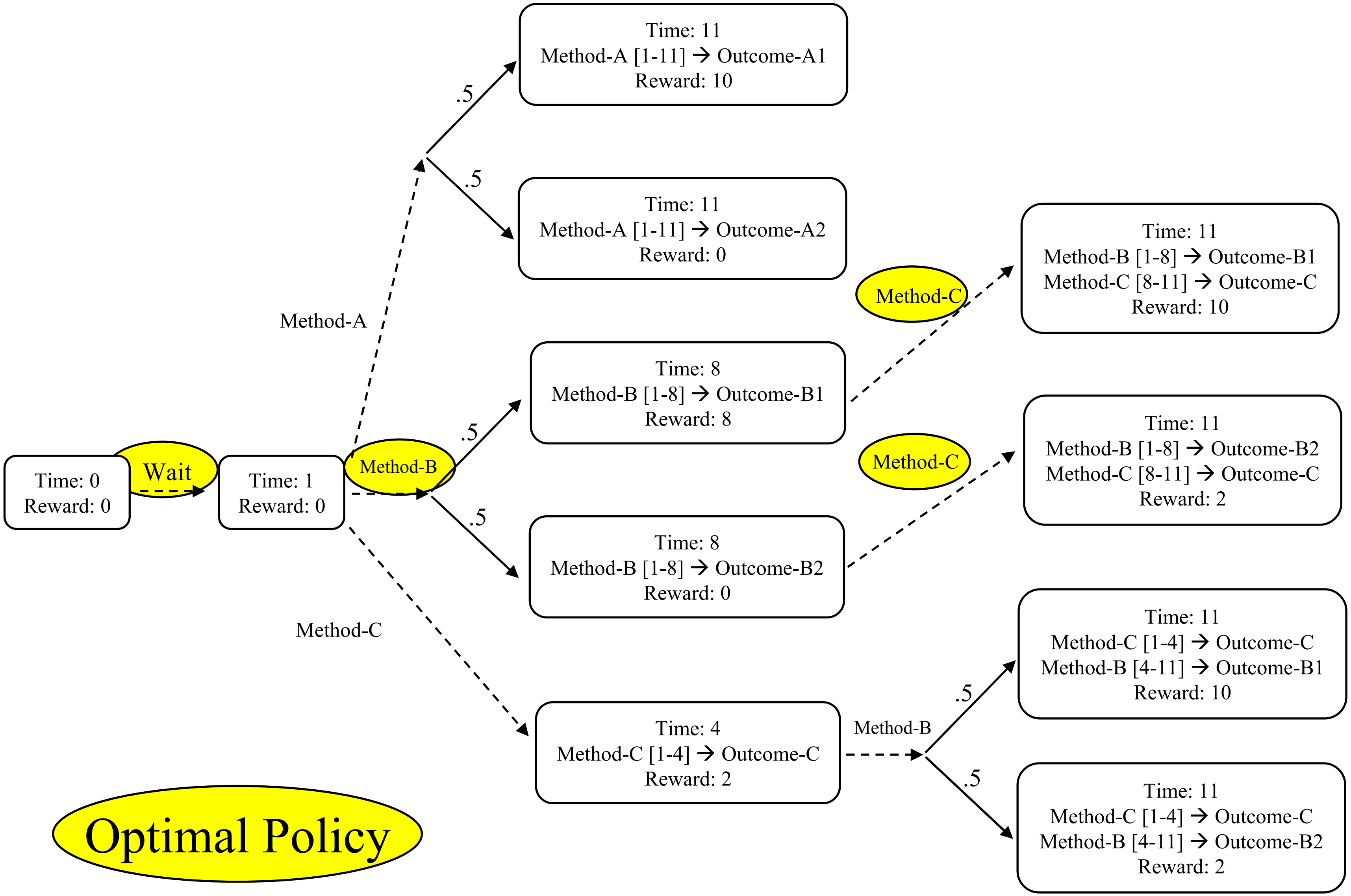
Mapping TAEMS to MDPs

- MDP states represent possible future states of the world, where some methods have been executed and resulted in various outcomes.
- To achieve the Markov property, states will represent:
  - The current time.
  - What methods have been executed, and their outcomes.
- Actions in the MDP will correspond to method choices.
- The transition model will represent the possible outcomes for each method.
- For efficiency, many states with just time-increment differences are omitted (no loss of precision).
- We also currently omit ‘abort’ action choice at all times except method deadline.

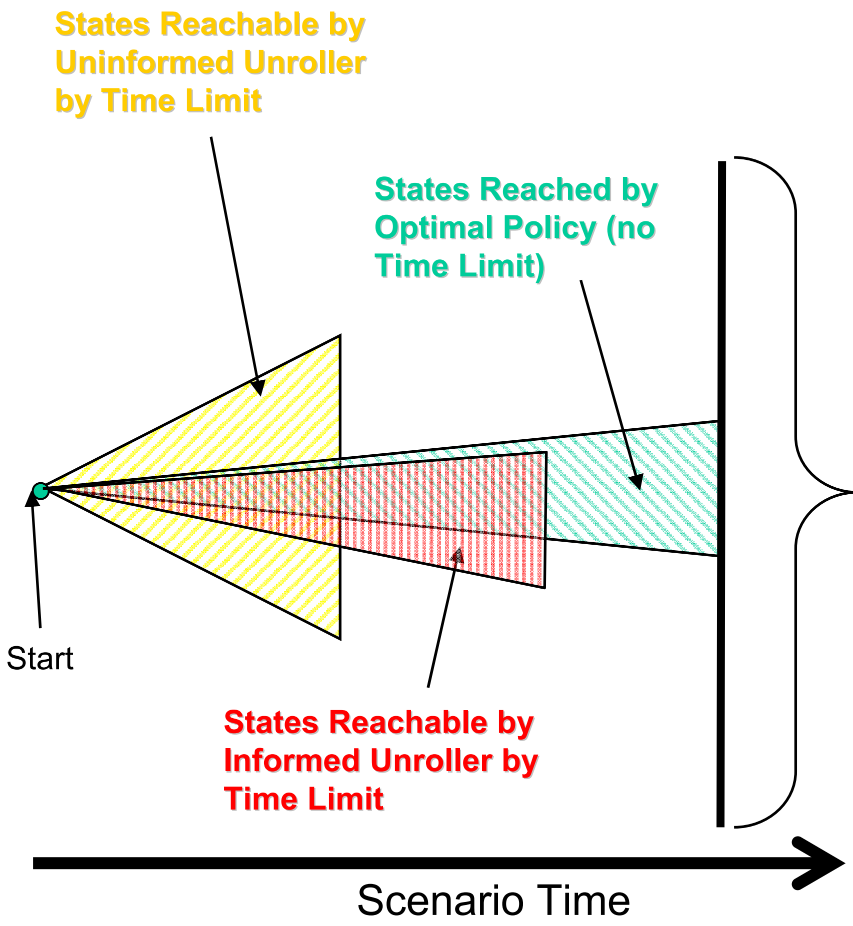
Simple Single-Agent TAEMS Problem



Unrolled MDP



Informed (Directed) MDP Unroller



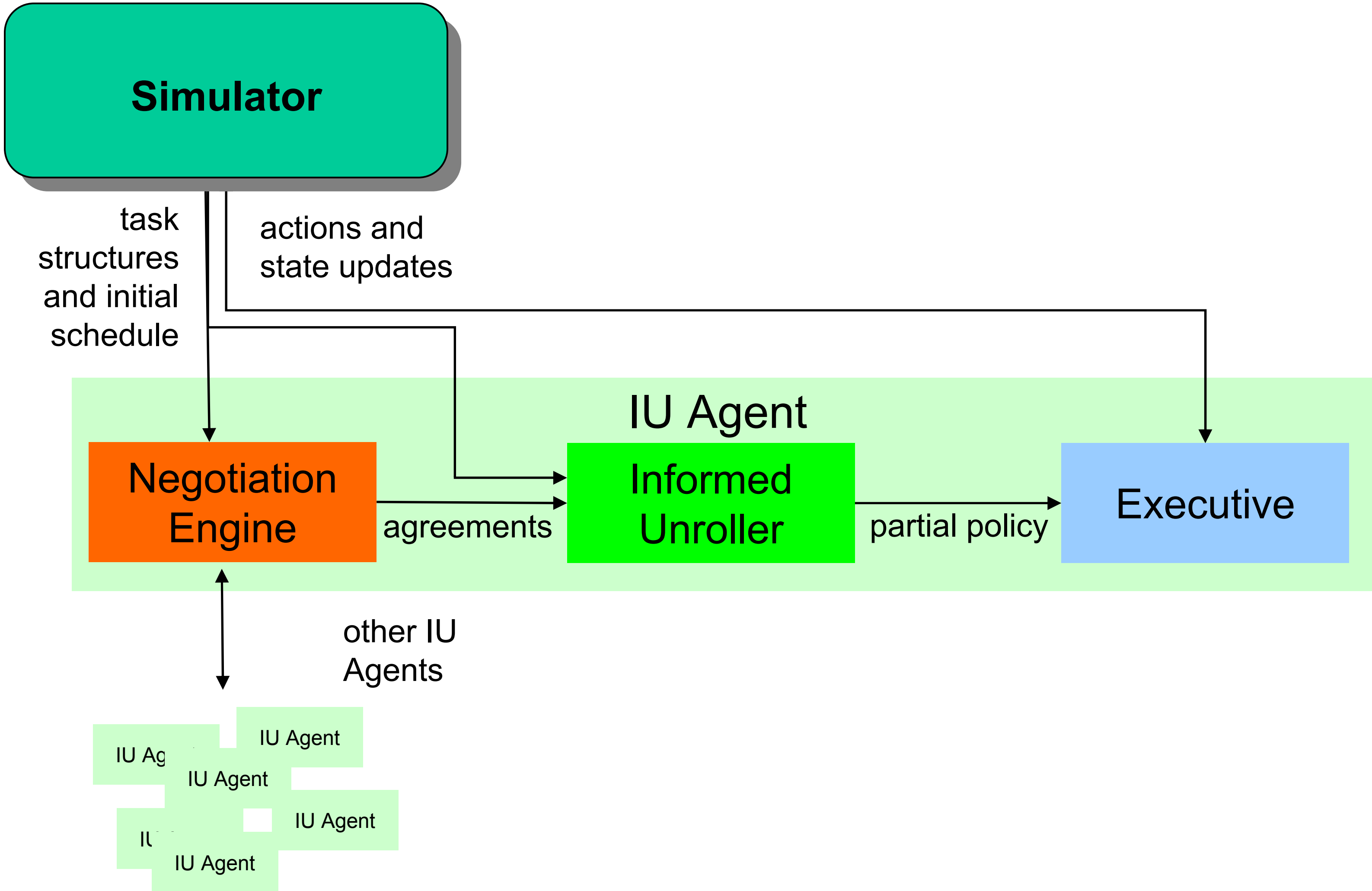
- Formulating real-world problems in an MDP framework often lead to a large state spaces.
- When computational capability is limited, we might be unable to explore the entire state space of an MDP.
- The decision about the subset of an MDP state space to be explored ("unrolled") affects the quality of the policy.
- Uninformed exploration can unroll to a particular time horizon.
- Informed exploration biases expansion towards states that are believed to lie along trajectories of high-quality policies.

Steering Local MDPs Towards Inter-Agent Coordination

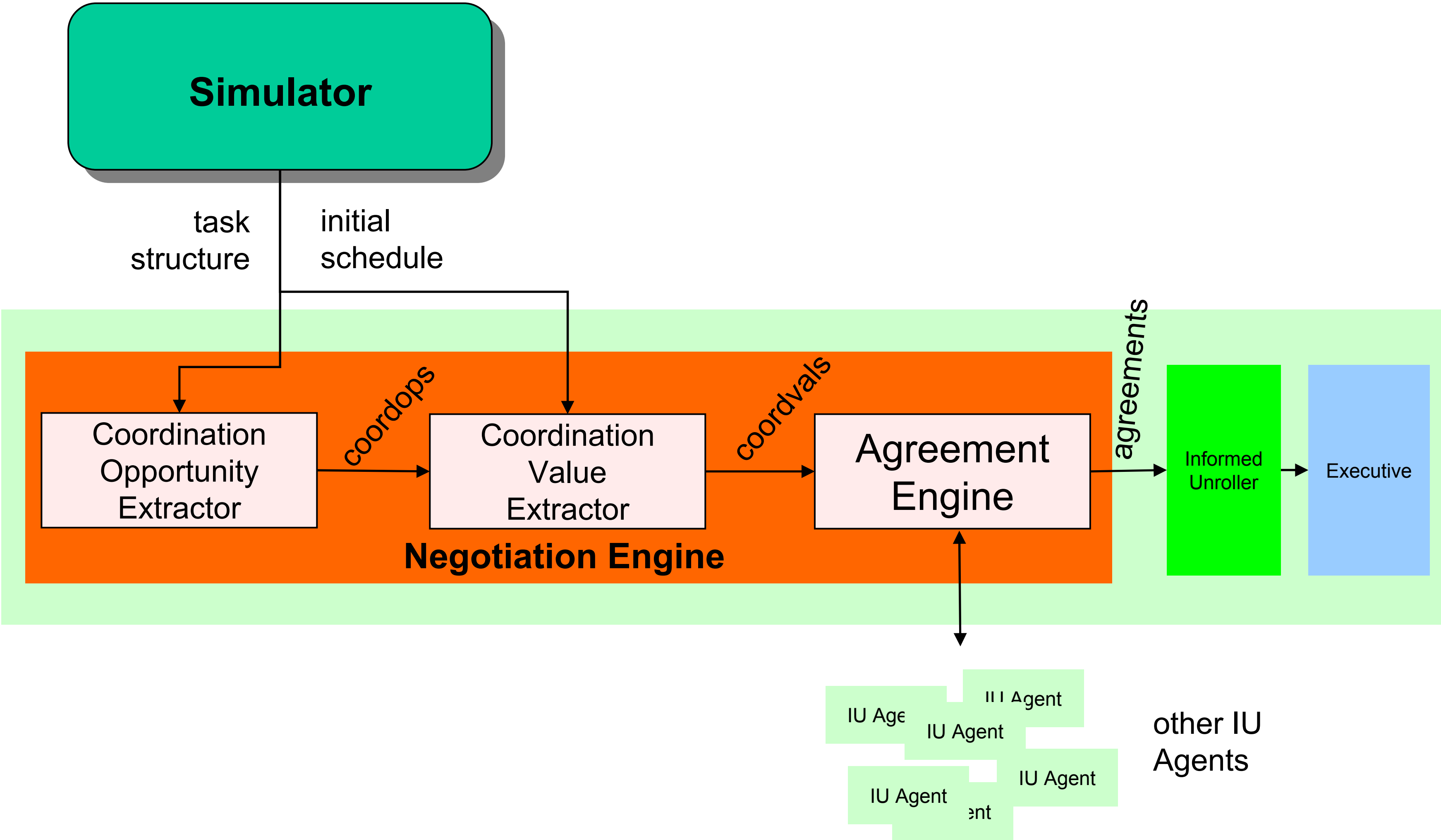
- MDPs require reward model to optimize.
- Assume local quality is a reasonable approximation to global quality.
  - This is not necessarily true.
  - In fact, some structures in CTAEMS make this dramatically incorrect.
    - E.g., SYNCSUM; semantics of surprise.

- Use communication to construct agreements over commitments.
- Use commitments to bias local MDP model to align local quality measures with global.

IU Agent Architecture



Negotiation Engine

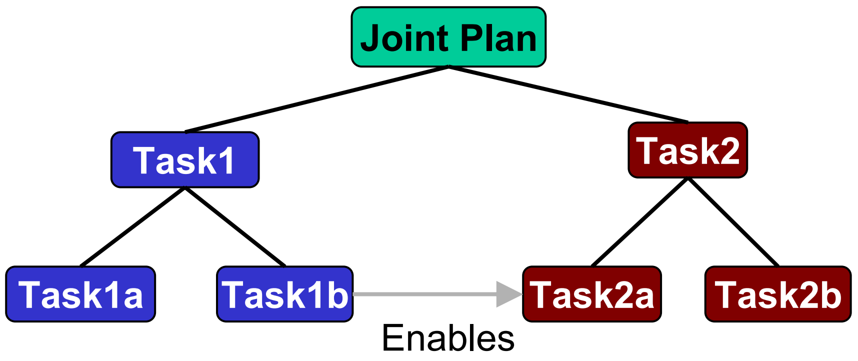


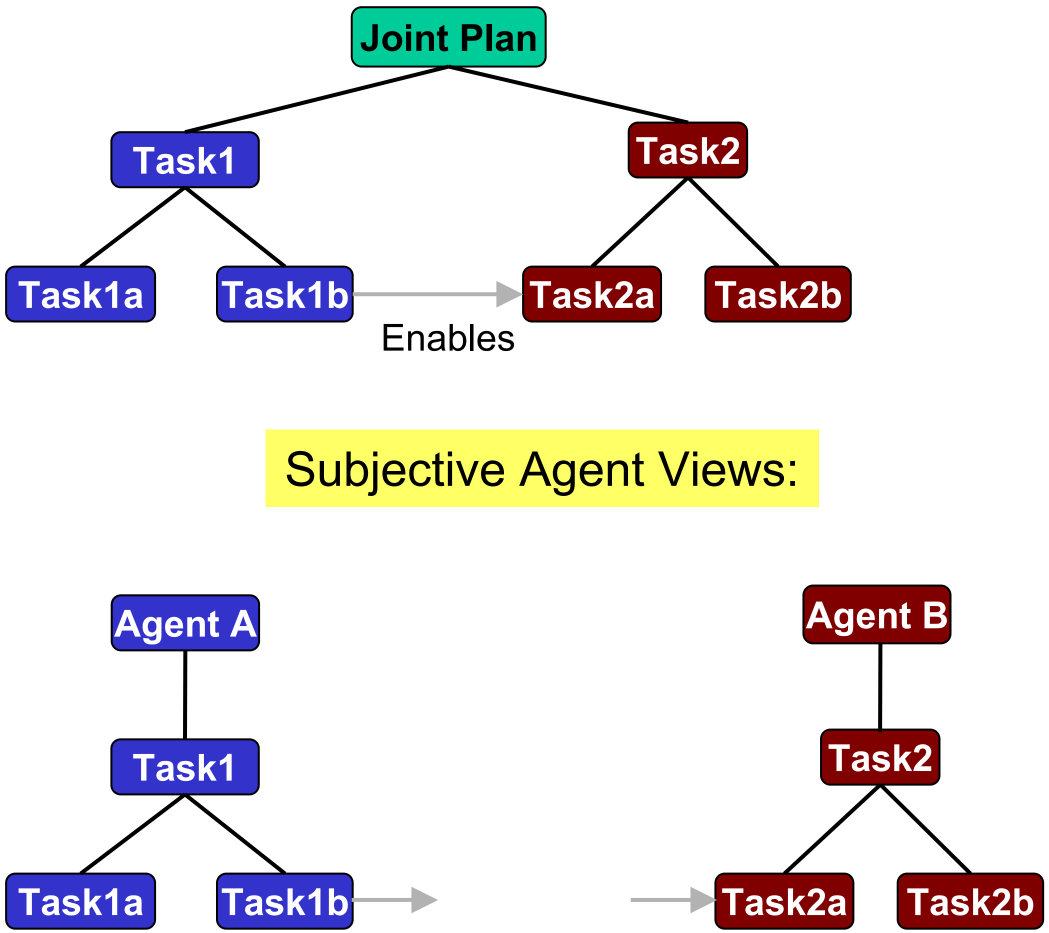
IU-Agent Control Flow Outline

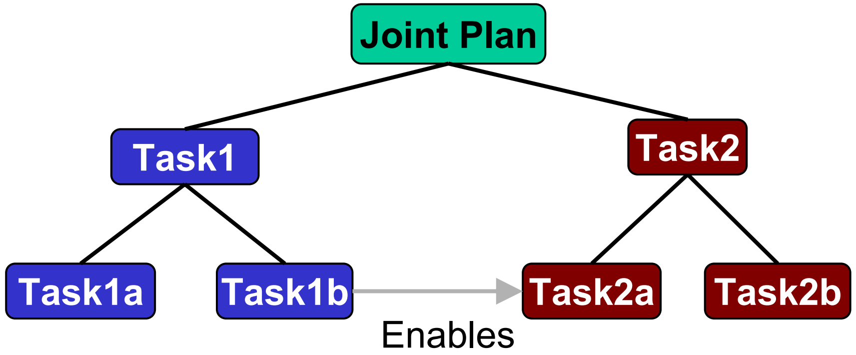
- Coordination opportunities identified in local TAEMS model (subjective view).
- Initial coordination value expectations derived from initial schedule.
- Communication establishes agreements over coordination values.
- Coordination values used to manipulate subjective view and MDP unroller, to bias towards solutions that meet commitments.
- Unroller runs until first method can be started. Derives partial policy.
- Executive runs MDP policy.
- If agent gets confused or falls off MDP, enters greedy mode.

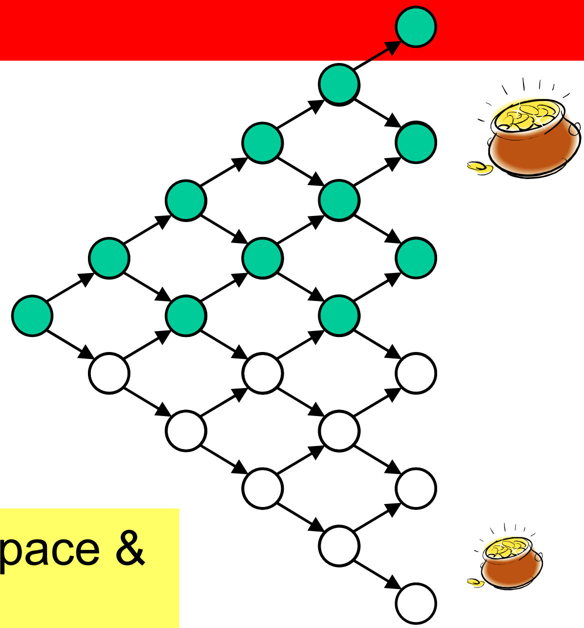
Steering MDP Policy Construction Towards Coordination

- Two primary ways of guiding MDP policies:
  - Additional reward or penalty attached to states with a specific property (e.g., achievement of quality in an enabling method by a specified deadline).
  - "Nonlocal" proxy methods representing the committed actions of others (e.g., synchronized start times).

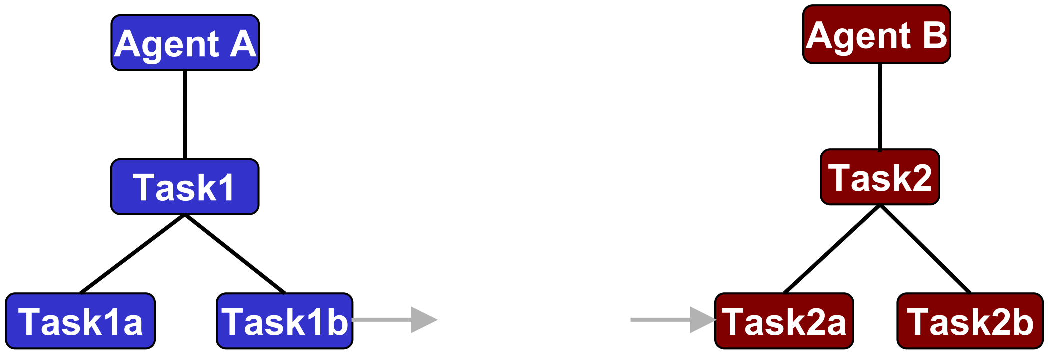

Joint Plan
Task1
Task2
Task1a
Task1b
Task2a
Task2b
Enables

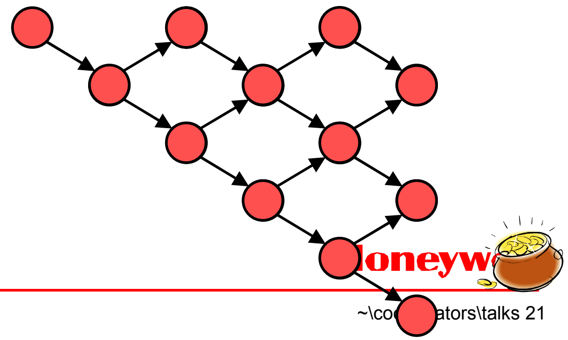

Joint Plan
Task1
Task2
Task1a
Task1b
Task2a
Task2b
Enables
Subjective Agent Views:
Agent A
Task1
Task1a
Task1b
Agent B
Task2
Task2a
Task2b

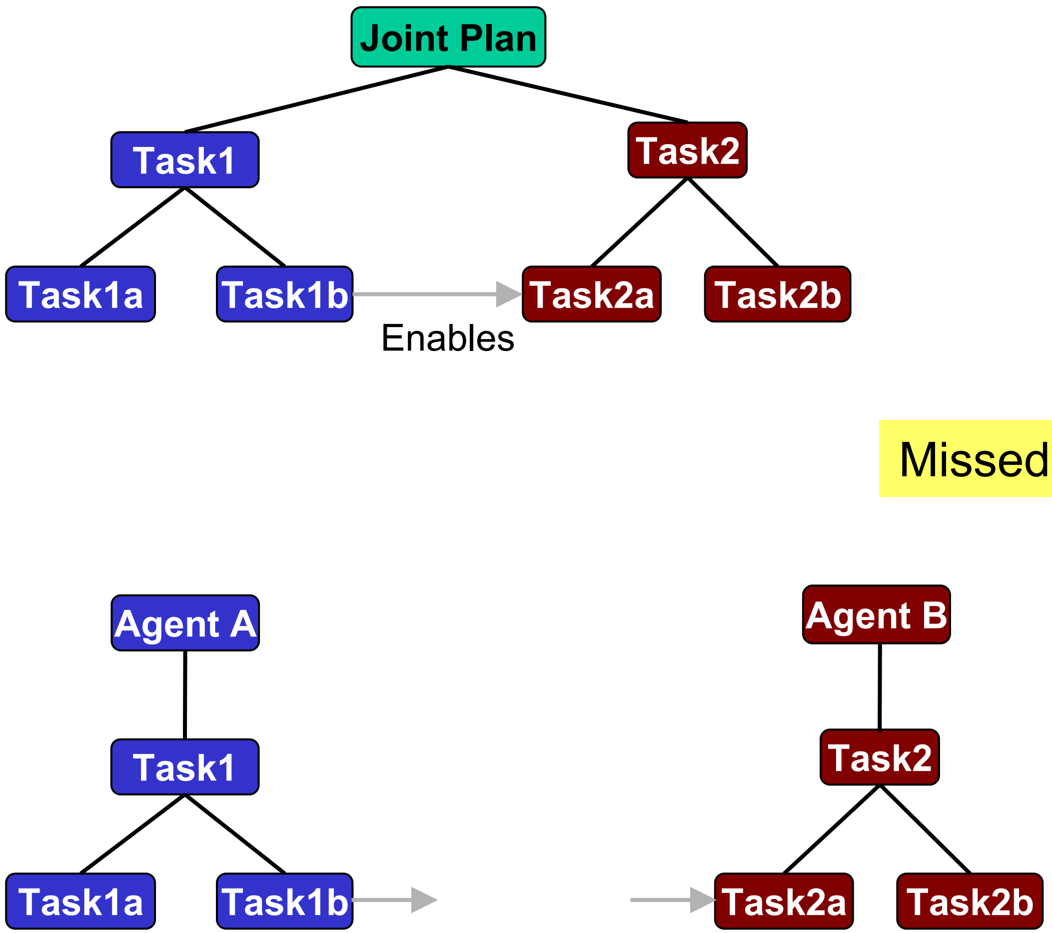




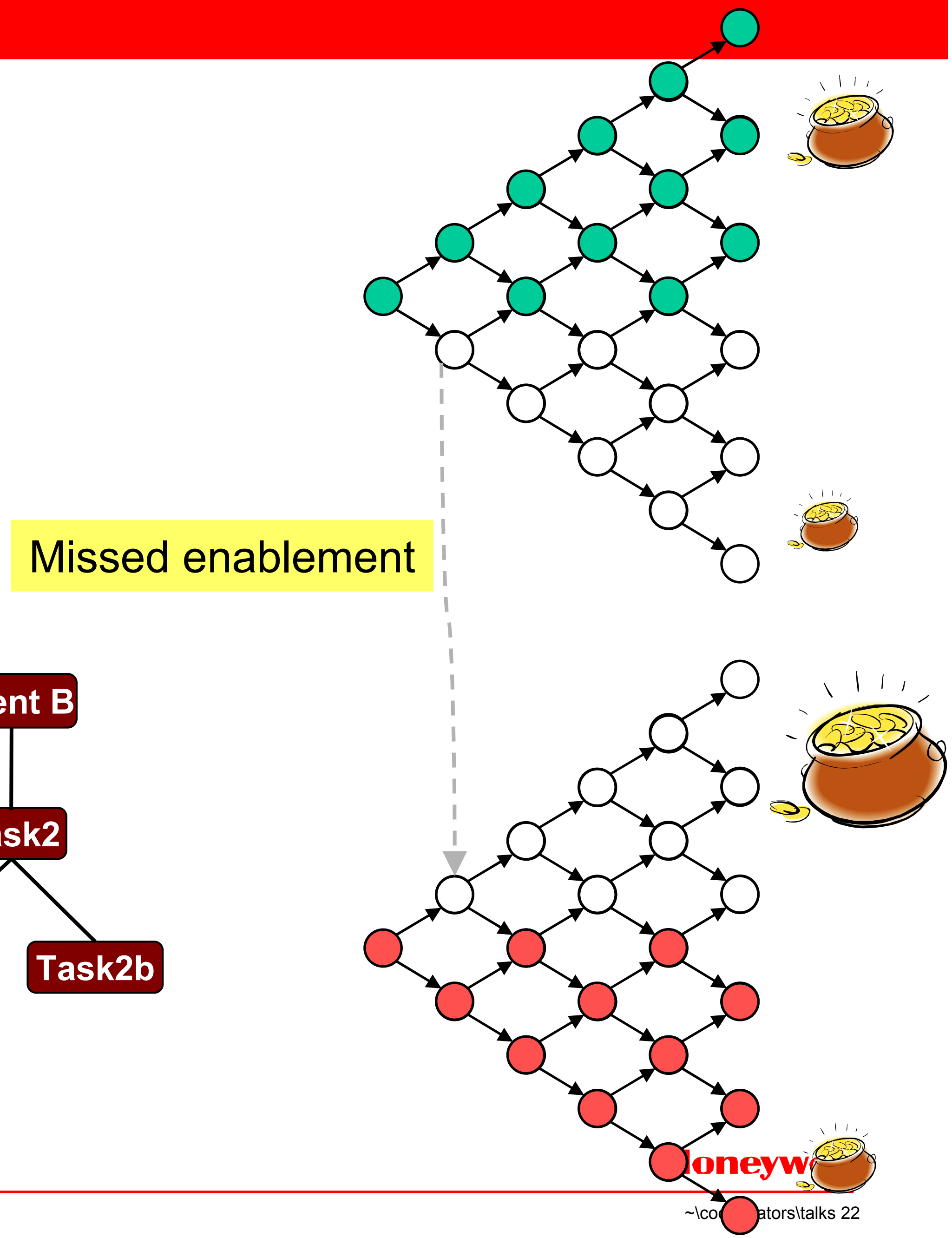
Corresponding MDP State Space &
Uncoordinated Policies



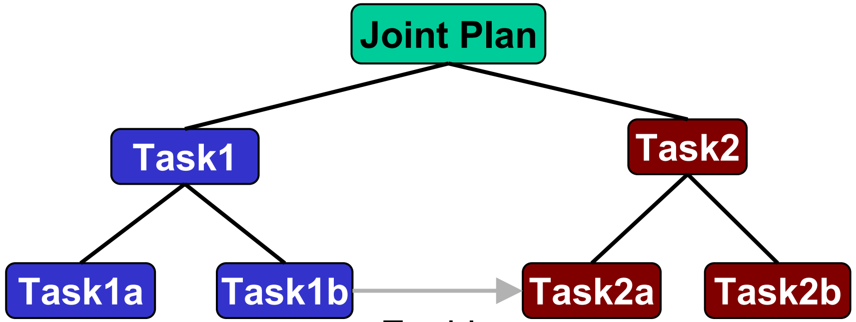




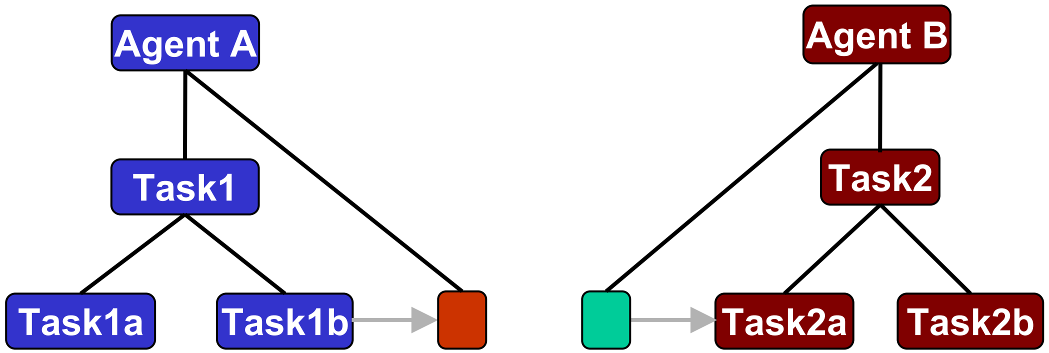
Joint Plan
Task1
Task2
Task1a
Task1b
Task2a
Task2b
Enables
Agent A
Task1
Task1a
Task1b
Agent B
Task2
Task2a
Task2b

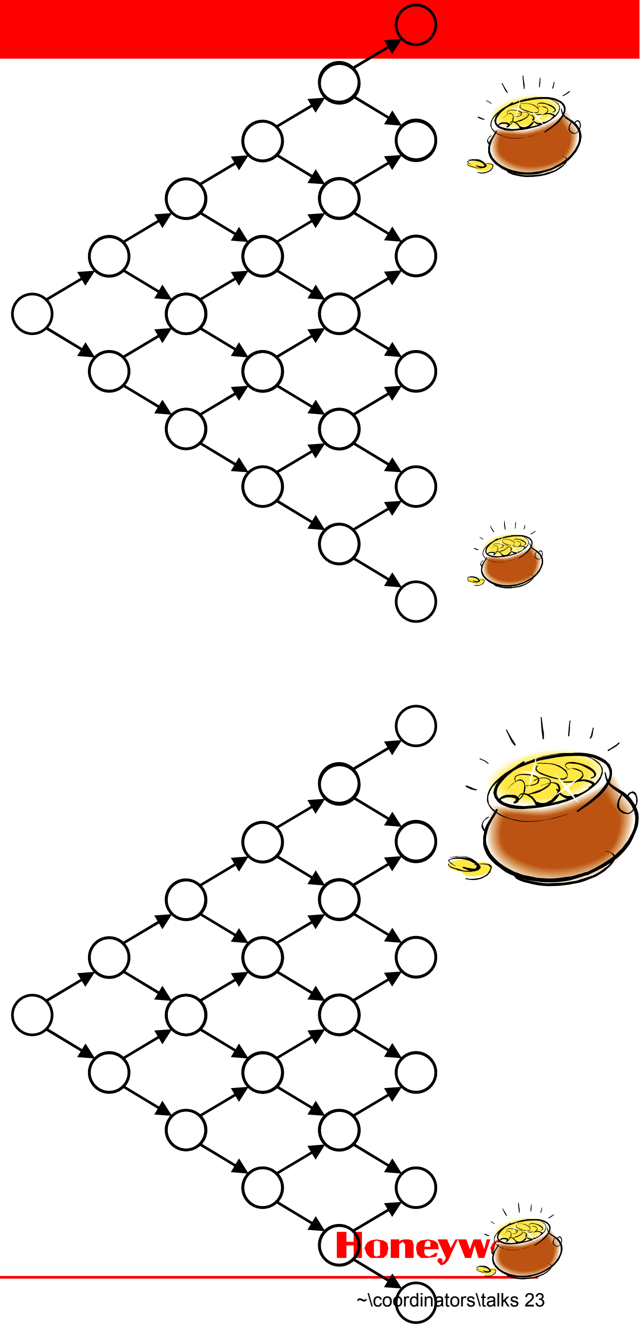


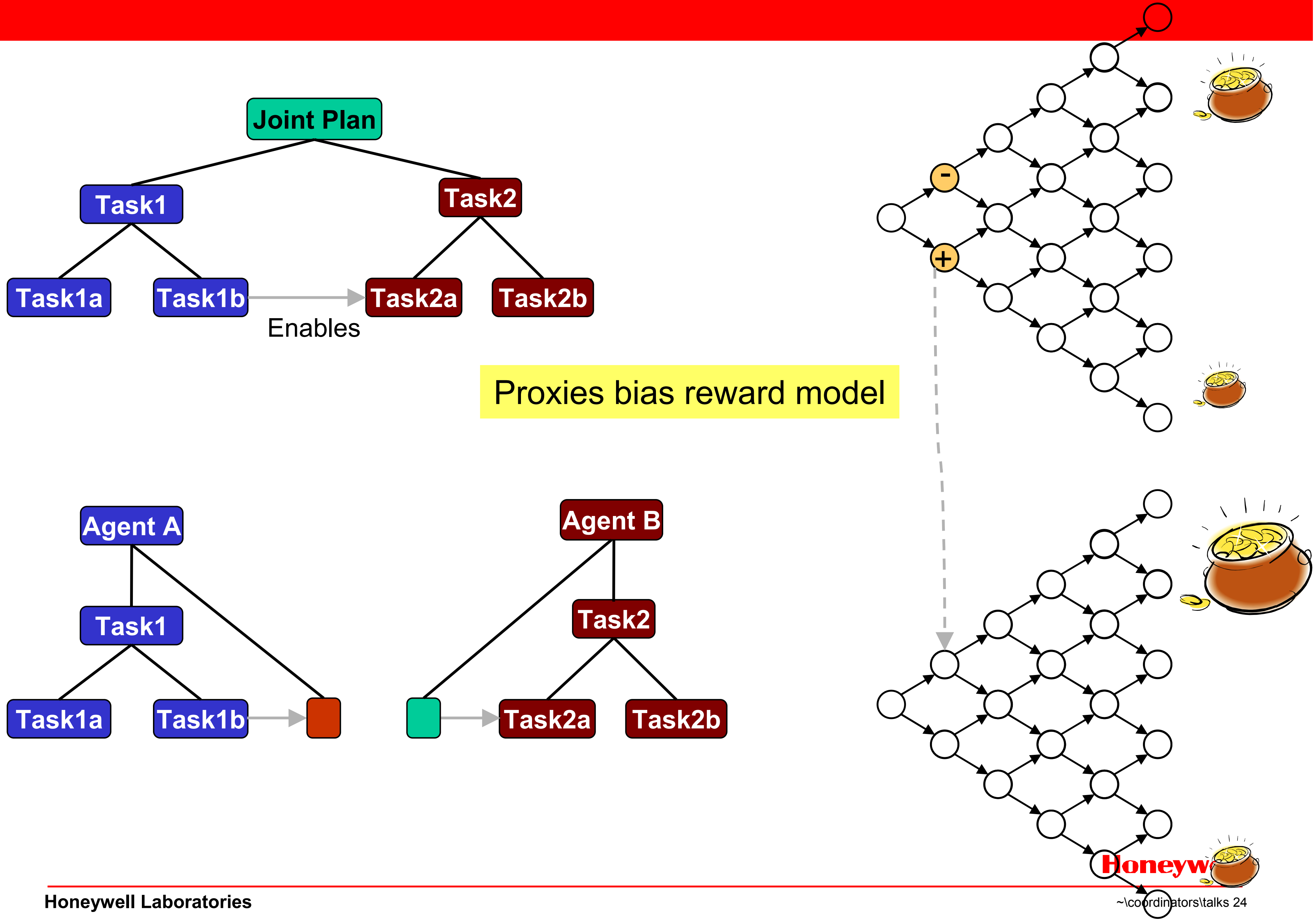
Missed enablement

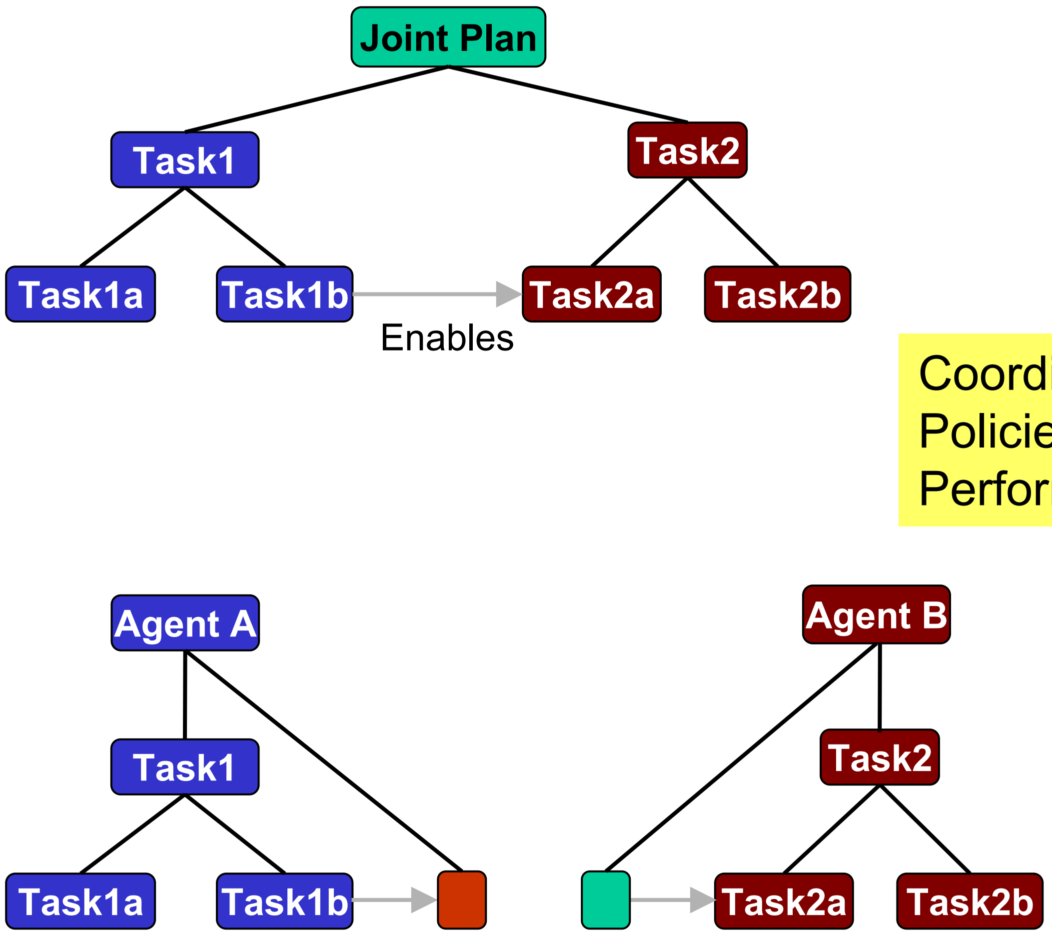


Proxy tasks represent
Coordination commitments.

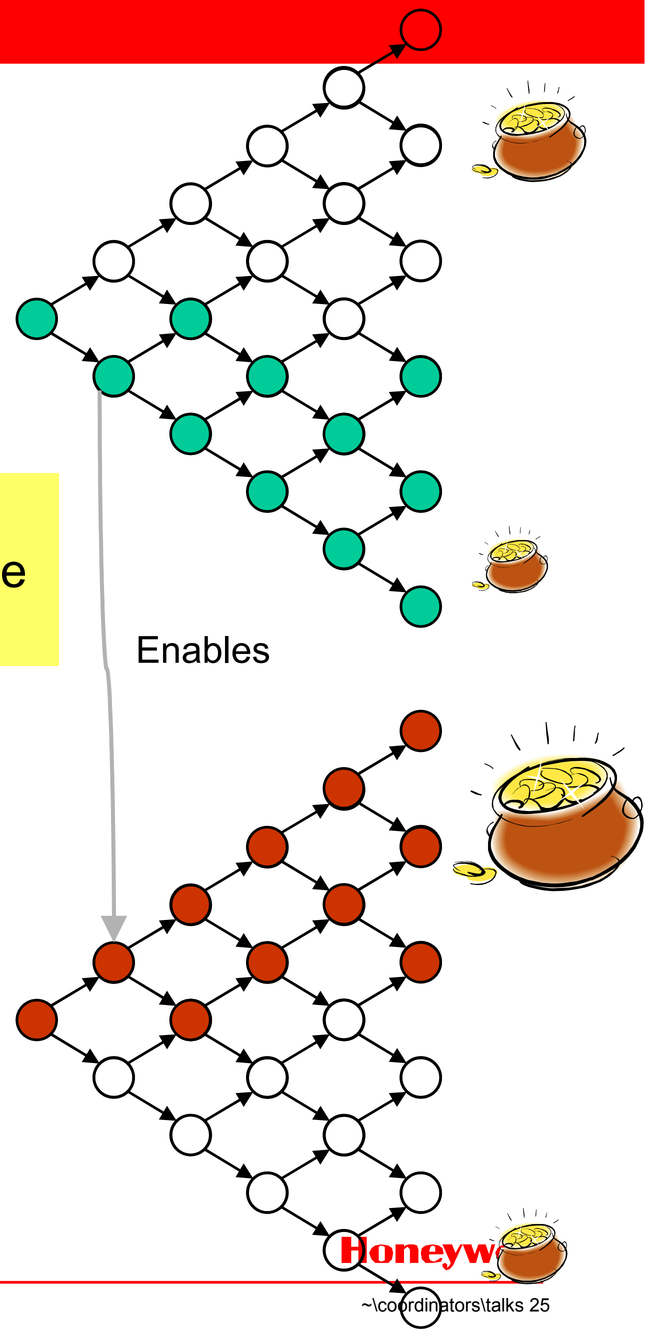





Joint Plan
Task1
Task2
Task1a
Task1b
Task2a
Task2b
Enables
Proxies bias reward model
Agent A
Agent B
Task1
Task2
Task1a
Task1b
Task2a
Task2b

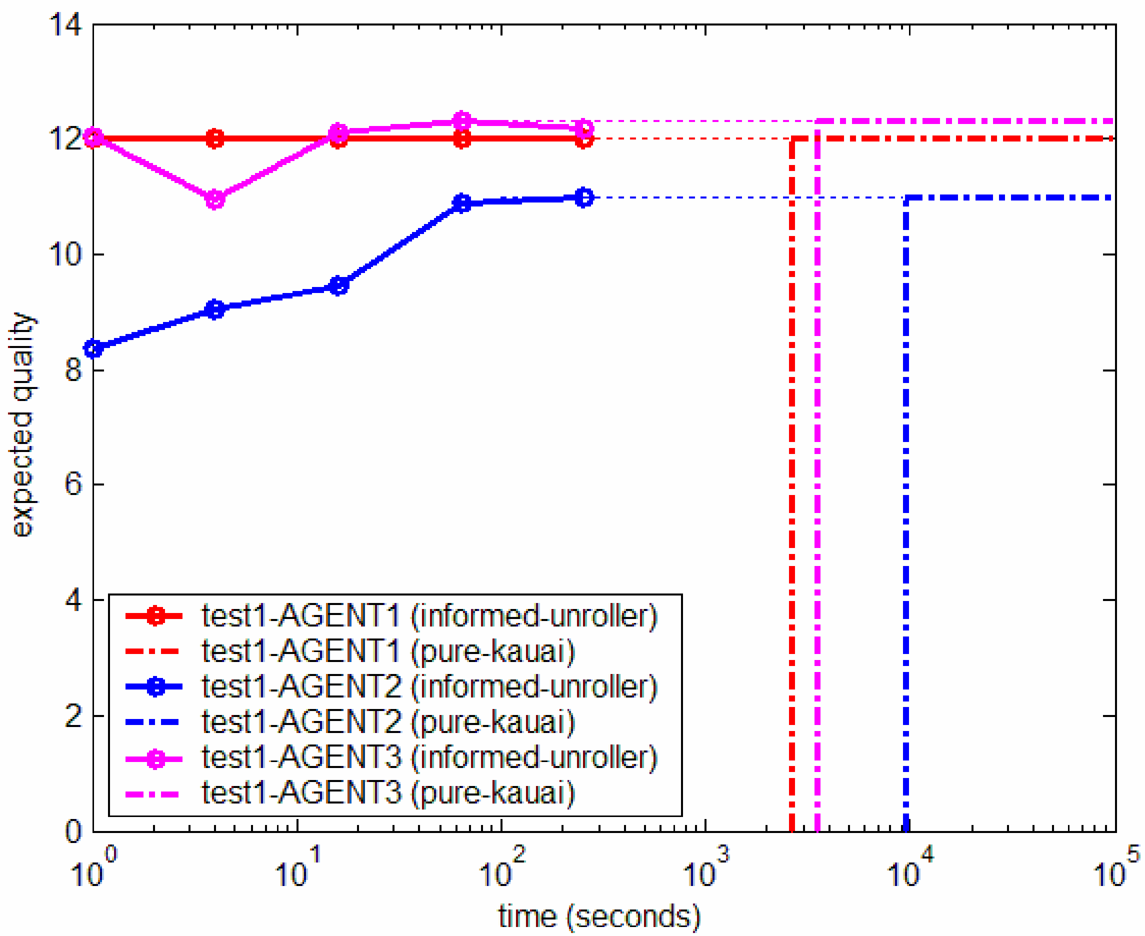


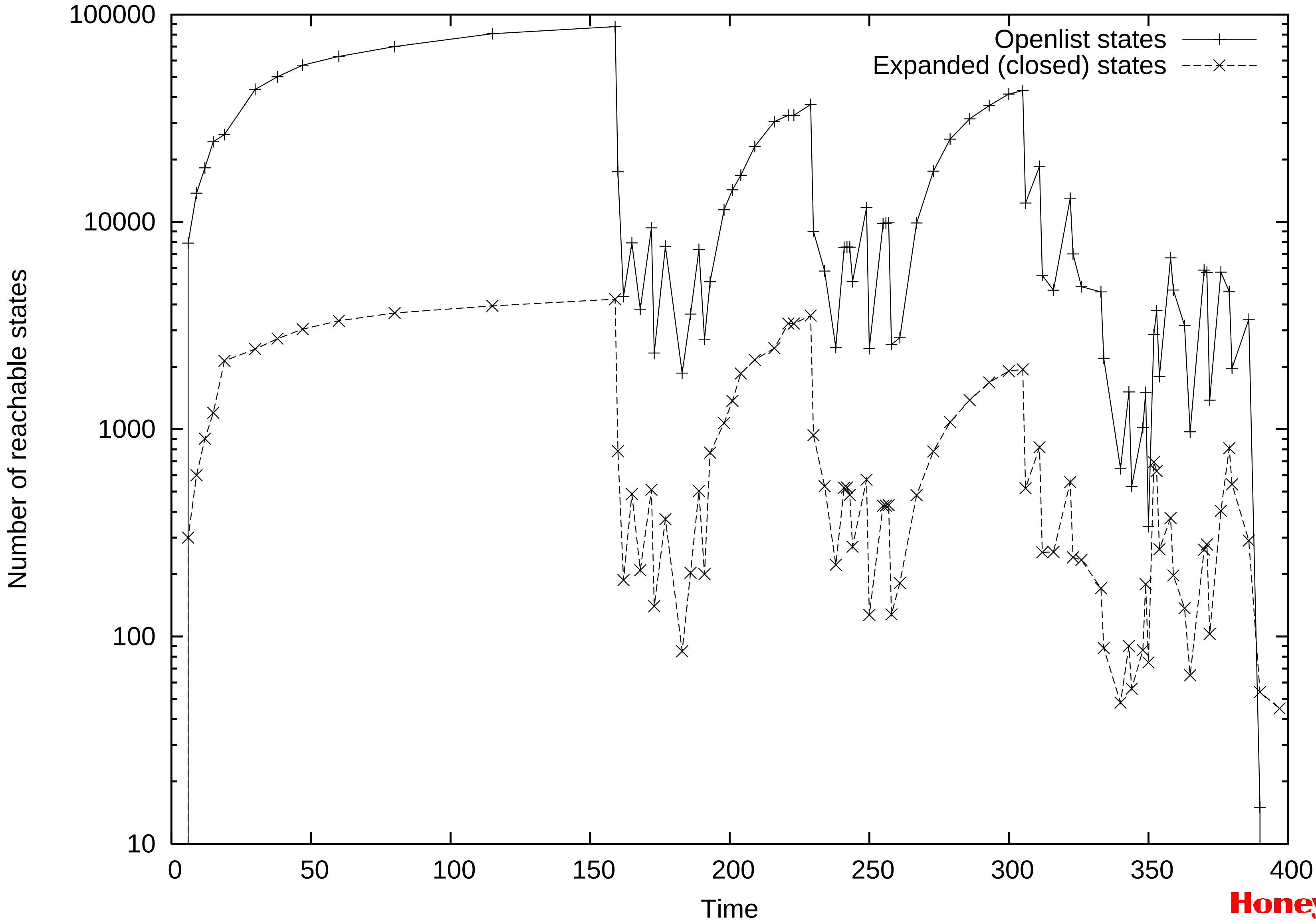
Coordinated Policies Improve Performance



Informed Unroller Performance

- Anytime.
- Converges to optimal complete policy.
- Can capture bulk of quality with much less thinking.





100000
10000
1000
100
10
0
50
100
150
200
250
300
350
400
Number of reachable states
Time
Openlist states
Expanded (closed) states

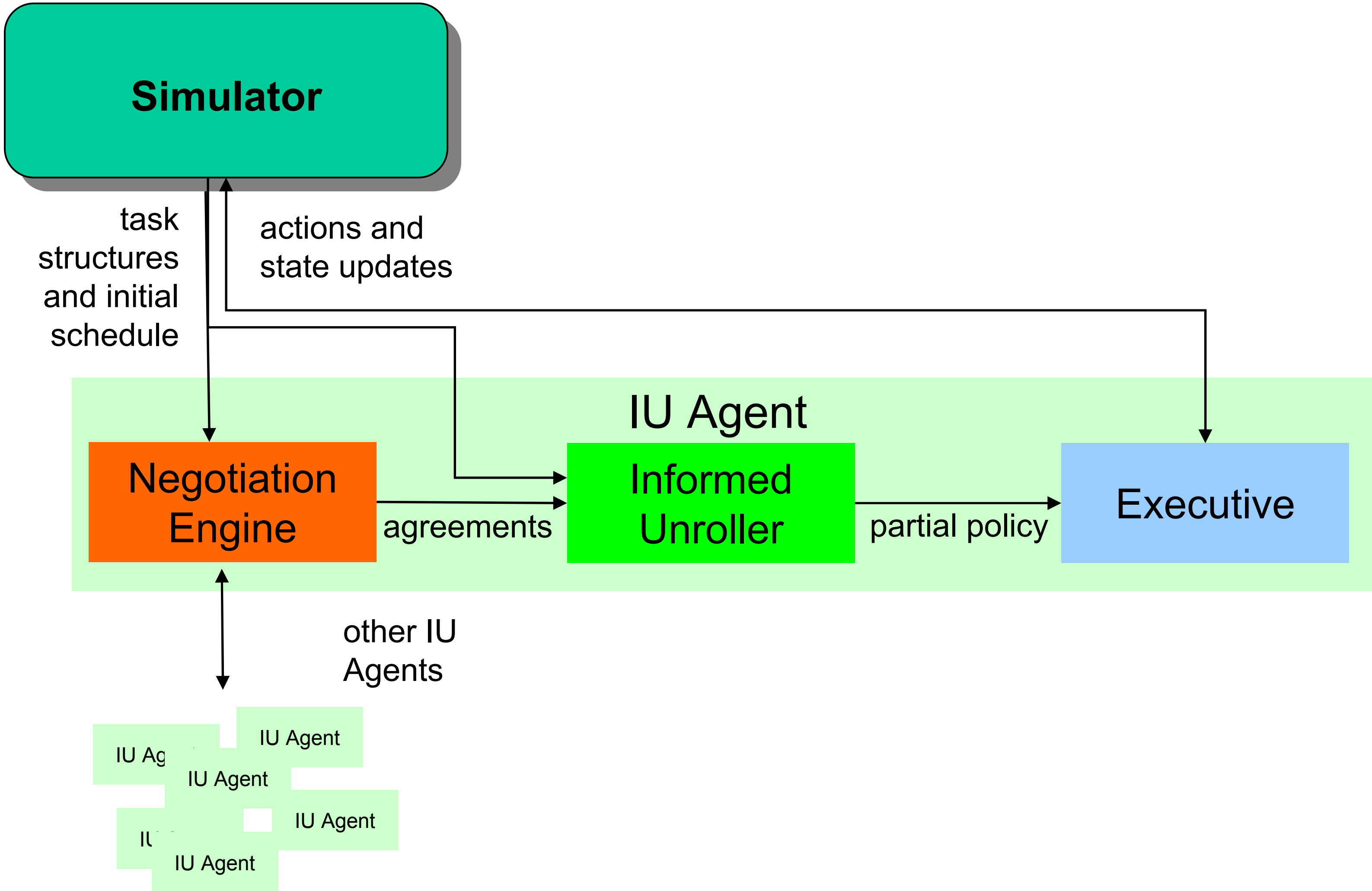
Honeywell

Lessons and Future Directions

- Integration of the deliberative and reactive components is challenging (as always).
  - The IU-Agent may be the first embedded online MDP-based agent for complex task models.
- Pruning based on runtime information is critical to performance.
- Meta-control is even more critical:
  - When to stop increasing state space size to derive a policy based on space unrolled so far?
  - How to bias expansion: depth-first vs. breadth-first, as expanded horizon and next-action-opportunity time varies.

THE END

IU Agent Architecture



Markov Decision Processes: What Are They?

- Formally-sound model of a class of control problems: what action to choose in possible future states of the world, when there is uncertainty in the outcome of your actions.
- State-machine representation of changing world, with:
  - Controllable action choices in different states.
  - Probabilistic representation of uncertainty in the outcomes of actions.
  - Reward model describing how agent accumulates reward/utility.
- Markov property: each state represents all the important information about the world; knowing what state you are in is sufficient to choose your next action. (No history needed)
- Optimal solution to an MDP is a ***policy*** that maps every possible future state to the action choice that maximizes *expected utility*.

Markov Decision Process Overview

- Model: A set of *states* (S) in which agent can perform subset of *actions* (A), resulting in probabilistic *transitions* (δ(s,a)) to new states and *reward* for each state and action (R(s,a)).
- Markov assumption: the next state and reward are only functions of the current state and action, no history required.
- Solution *policy* (π) specifies what action to choose in each state, to maximize expected lifetime reward.
- For infinite-horizon MDPs:
  – Use future-reward discount factor to prevent infinite lifetime reward.
  – Value/policy-iteration algorithms can find optimal policy.
- For finite-horizon MDPs, Bellman backup (dynamic programming) solves for optimal policy in $O(|S|^3)$ without reward discounting.
- Given a policy, can analytically compute expected reward (no simulation or sampling required).

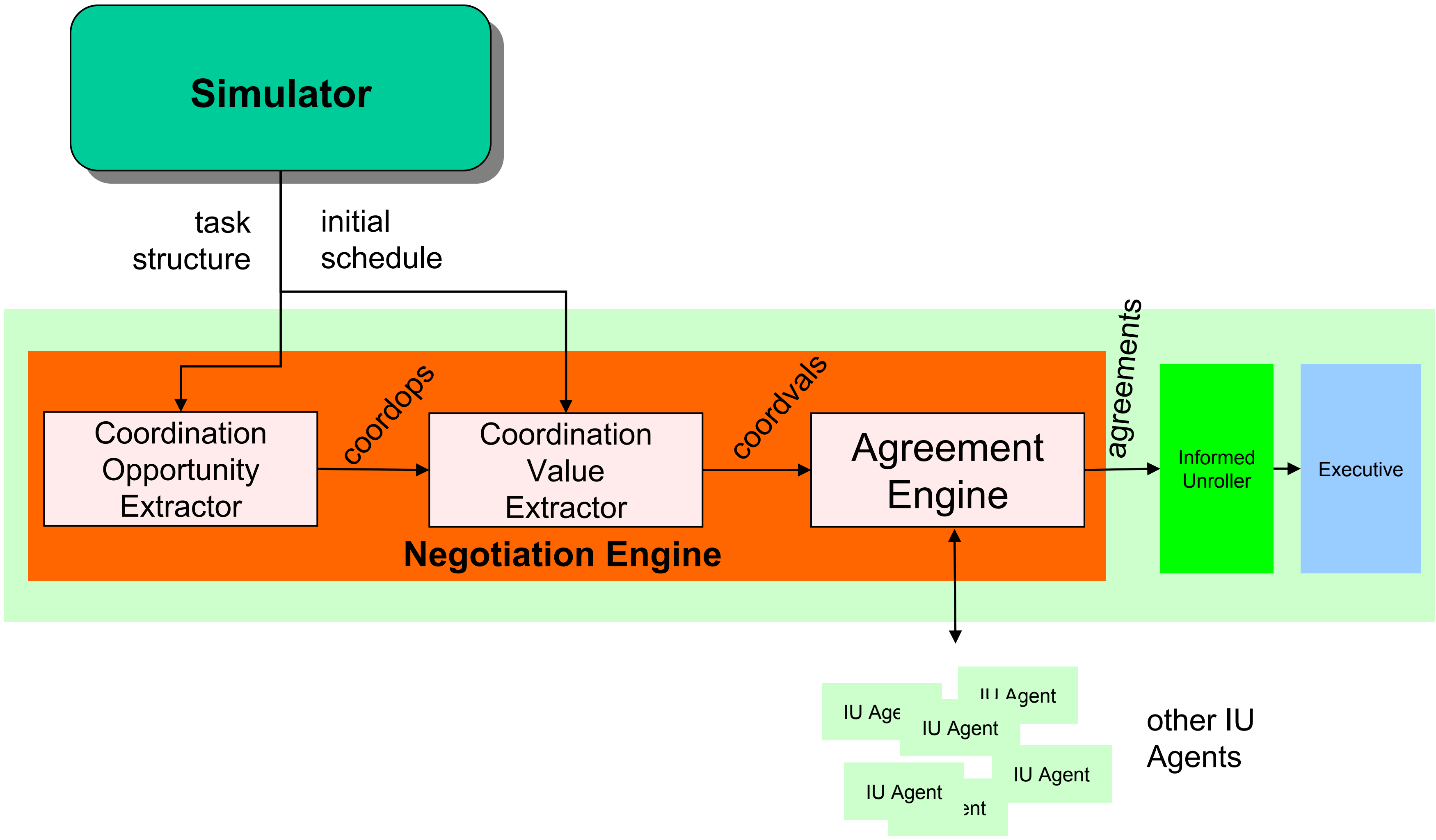
Why Use MDPs?

- Explicit representation of uncertainty.
  - Rationally balance risk and duration against potential reward.
  - TAEMS domains can include exactly this type of tradeoff (e.g., a longer method may achieve high quality or fail; a shorter method may be more reliable but yield lower quality).
- Accounts for delayed reward (e.g., from enabling later methods).
- Formal basis for defining optimal solutions.
  - When given an objective TAEMS multi-agent model, Kauai can derive an optimal policy if given enough time.
- Efficient existing algorithms for computing optimal policies.
  - Polynomial in the number of MDP states.
- Downside: state space can be very large (exponential).
  - Multi-agent models are even larger.

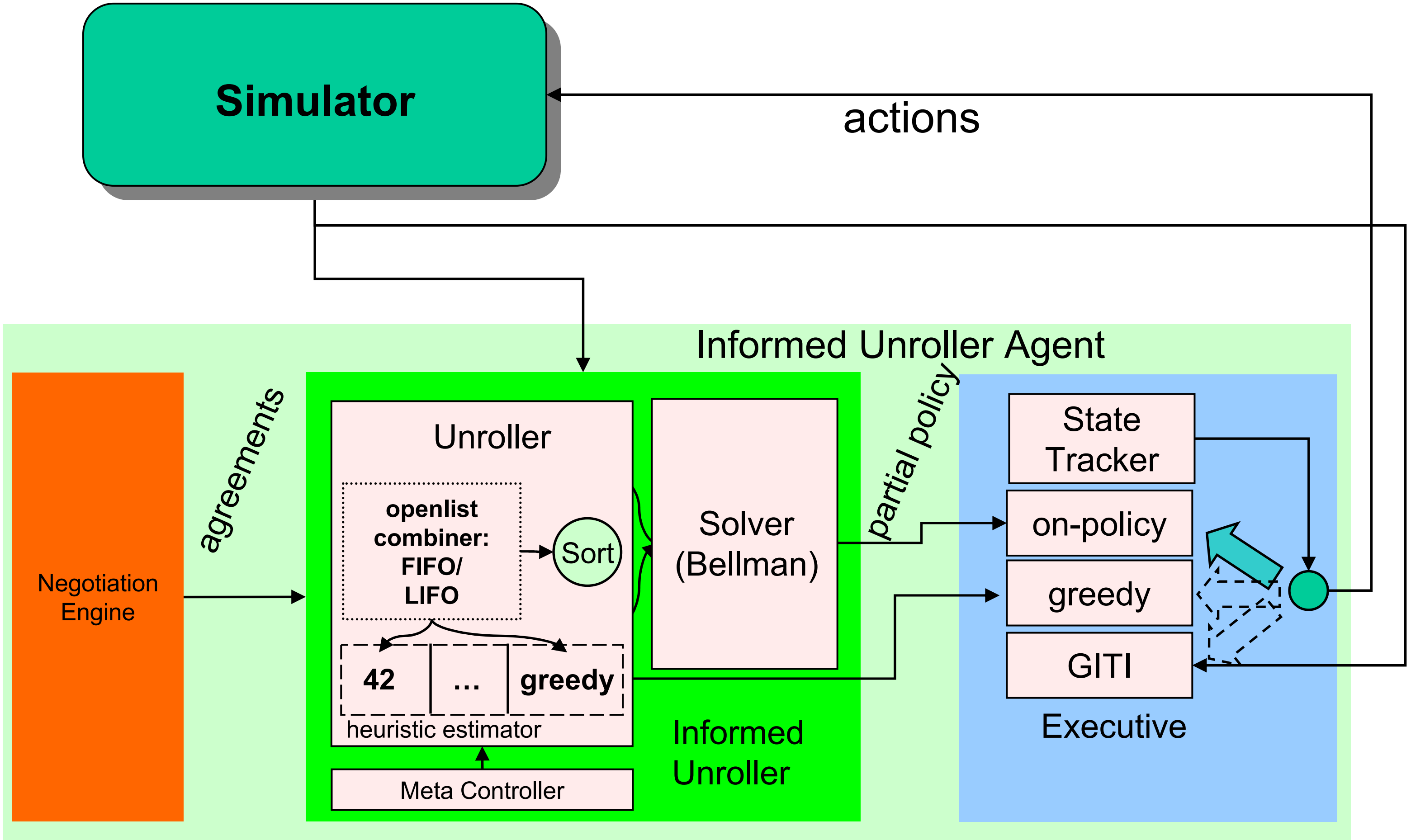
Domains Where MDPs Should Dominate

- When predictions of future possible outcomes can lead to different action choices.
- Reactive methods which do not look ahead can get trapped in “garden path” dead-ends.
- End-to-end methods that do not consider uncertainty cannot balance risk and duration against reward.
- MDPs inherently implement two forms of ***hedging***:
  - Pre-position enablements to avoid possibility of failure.
  - Choose lower-quality methods now to ensure higher overall expected quality.
- Expectations about future problem arrivals (meta-TAEMS) can also influence MDP behavior and improve performance.

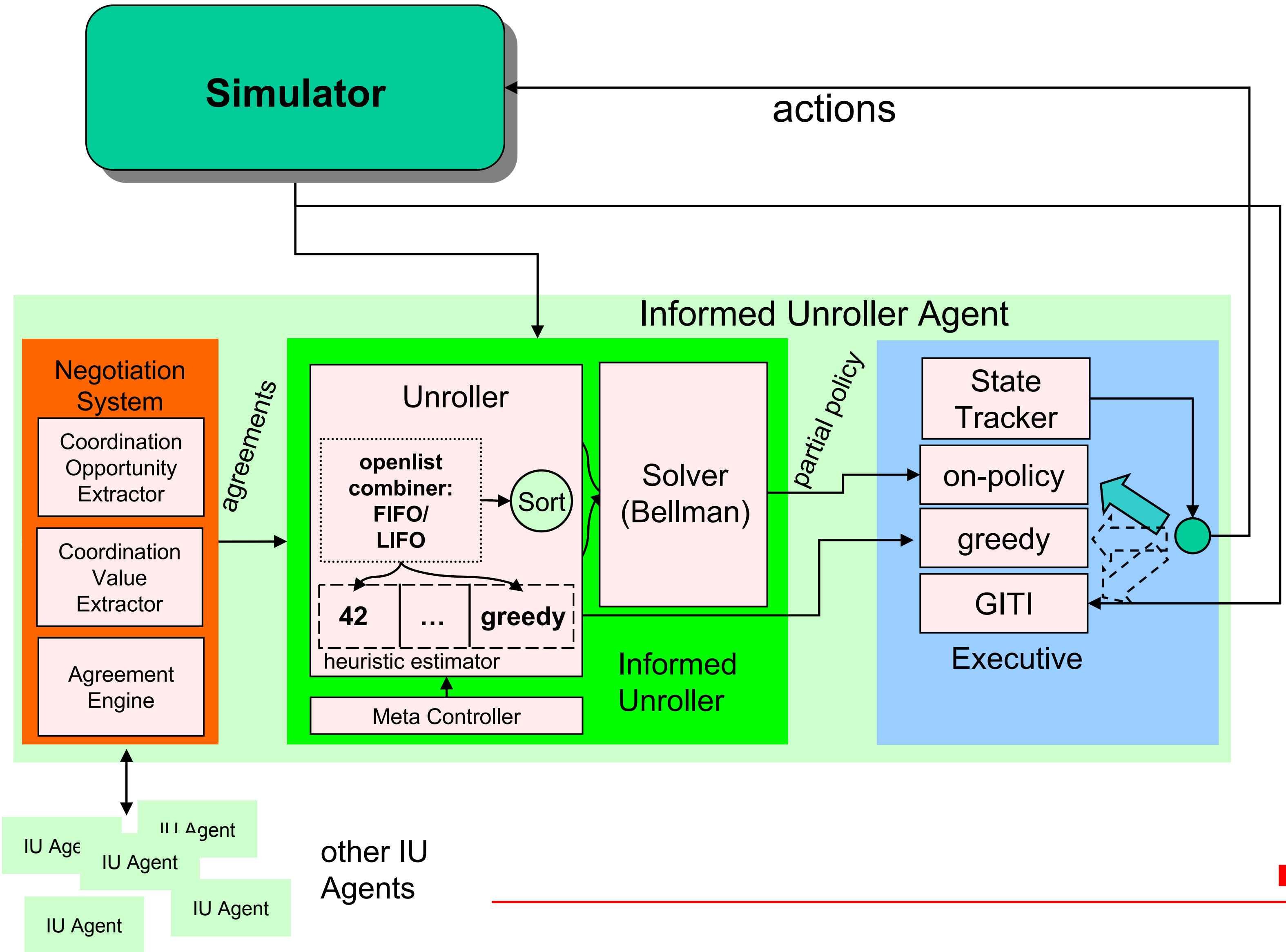
Negotiation Engine



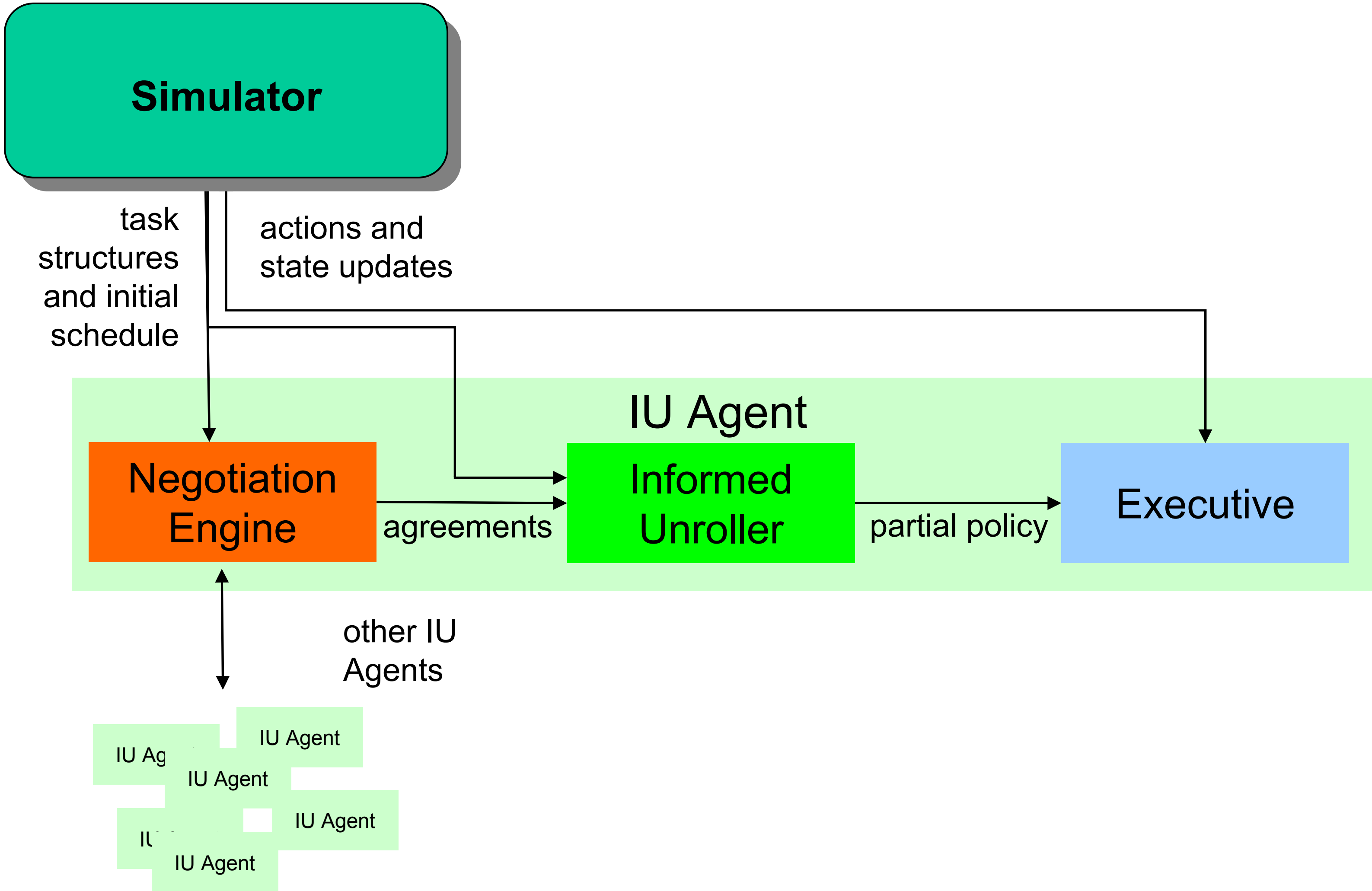
Informed Unroller and Executive



Informed Unroller and Executive



IU Agent Architecture

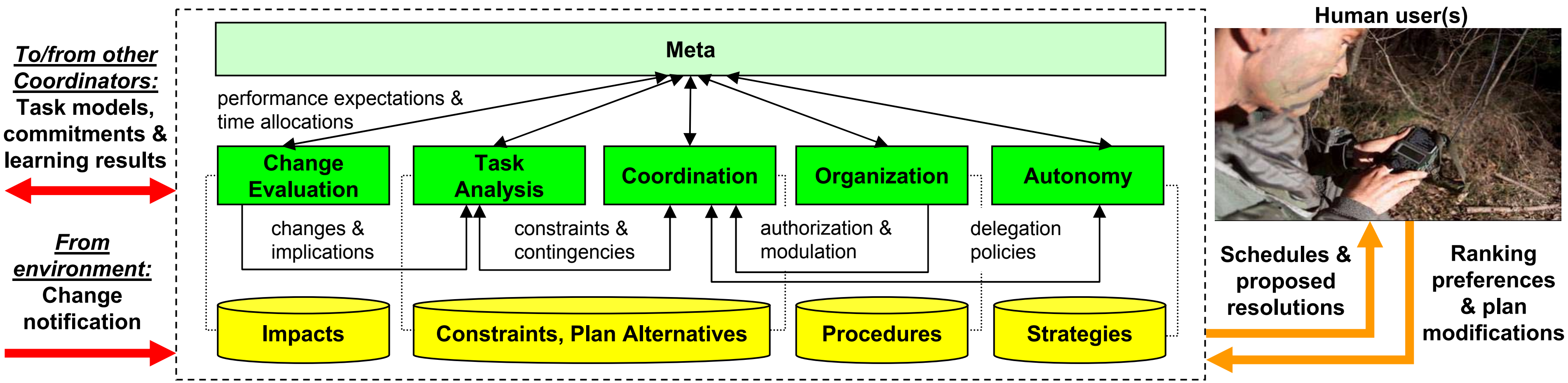


Motivating Problem

- Coordination of mission-oriented human teams, at various scales.
  - First responders (e.g., firefighters).
  - Soldiers.

- Distributed, multi-player missions.
- Complex interactions between tasks.

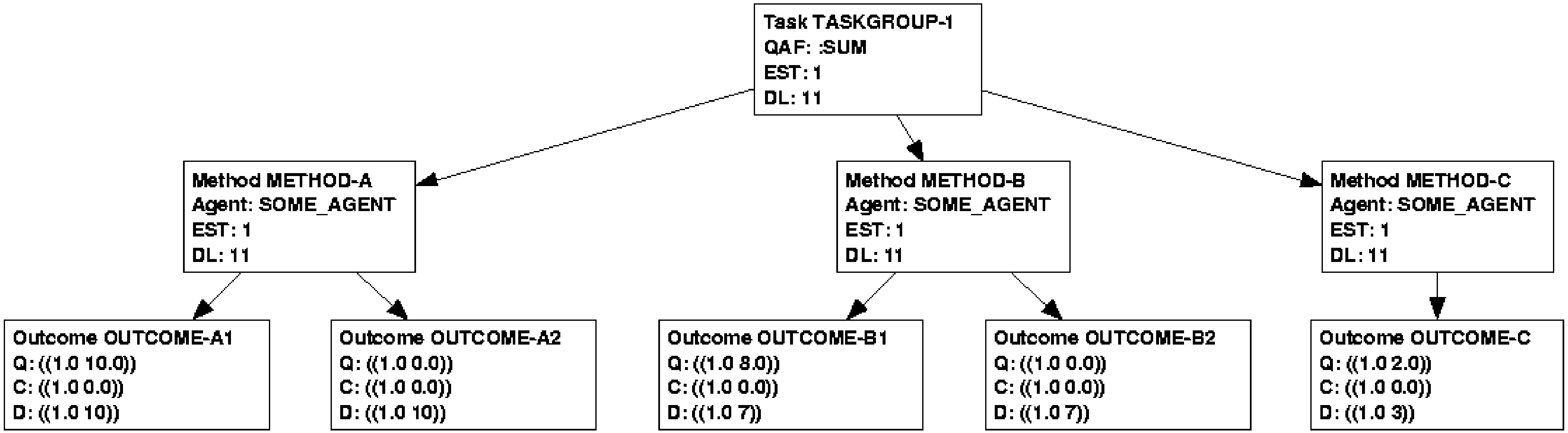
Architecture



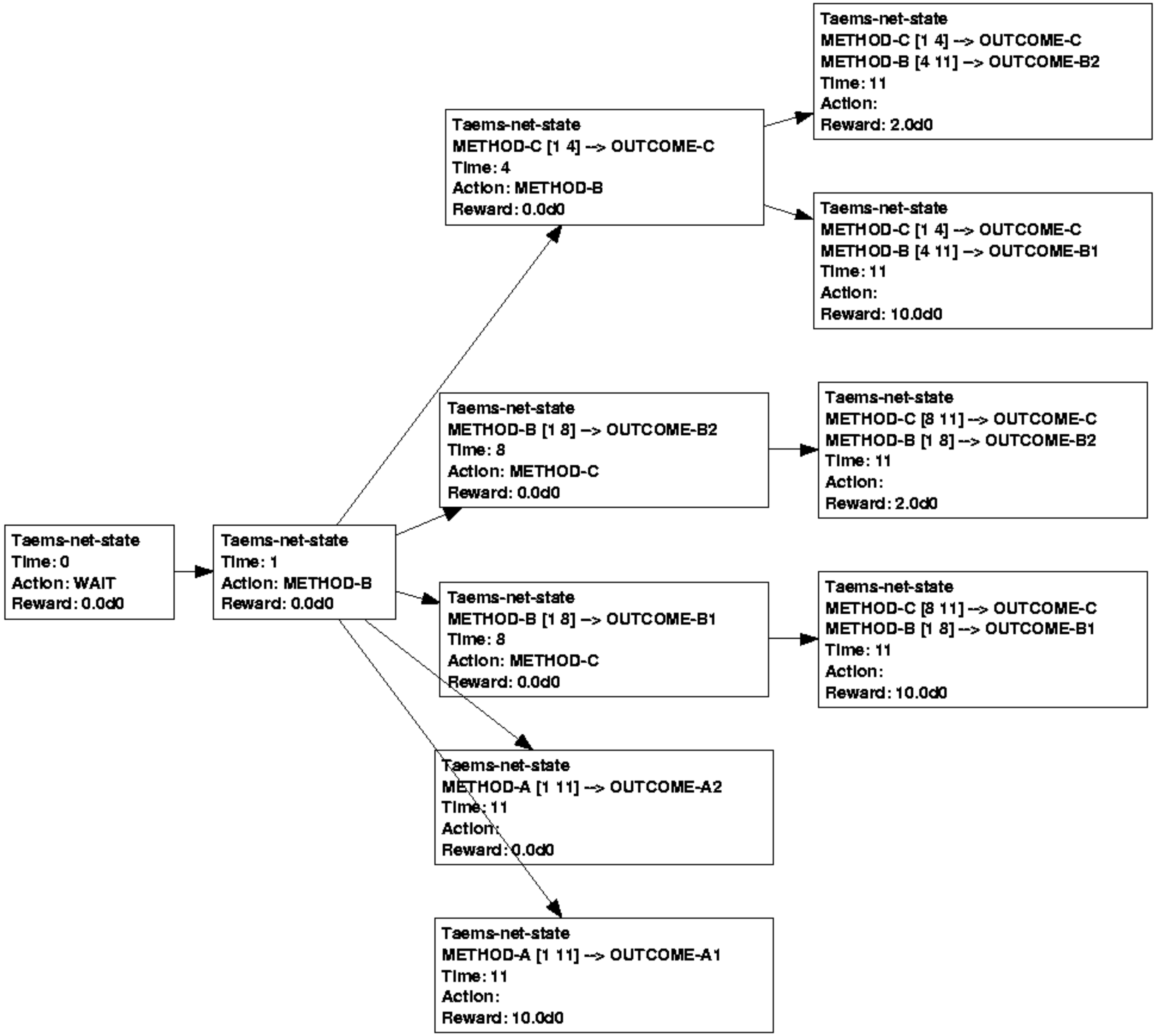
Mapping TAEMS to MDPs

- MDP states represent possible future states of the world, where some methods have been executed and resulted in various outcomes.
- To achieve the Markov property, states will represent:
  - The current time.
  - What methods have been executed, and their outcomes.
- Actions in the MDP will correspond to method choices.
- The transition model will represent the possible outcomes for each method.
- For efficiency, many states with just time-increment differences are omitted (no loss of precision).
- We also currently omit ‘abort’ action choice at all times except method deadline.
  - Pre-deadline aborts can be useful, but enormously expand state space.
  - Hope to remove/reduce this limitation in the future: can limit aborts to only when relevant times occur.

Simple Single-Agent TAEMS Problem



Unrolled MDP



IU-Agent Control Flow Outline

- Coordination opportunities identified in local TAEMS model (subjective view).
- Initial coordination value expectations derived from initial schedule.
- Communication establishes agreements over coordination values.
- Coordination values used to manipulate subjective view and MDP unroller, to bias towards solutions that meet commitments.
- Unroller runs until first method can be started. Derives partial policy.
- Executive runs MDP policy.
- If agent gets confused or falls off MDP, enters greedy mode.

Coordination Mechanism

- Local detection of possible coordination opportunities:
  - Enablement.
  - Synchronization.
  - Redundant task execution.
- Local generation of initial coordination values:
  - Use initial schedule to "guess" at good values.
- Communication
  - Establish that other agents are involved in coordinating:
    - Local information is incomplete.
    - Requires communication only among possible participants.
  - Establish a consistent set of coordination values:
    - Requires communication only among actual participants.

Steering MDP Policy Construction Towards Coordination

- MDP policies include explicit contingencies and uncertain outcomes.
- Enforcing a "guarantee" is frequently the wrong thing to do, because accepting a small possibility of failure can lead to a better *expected quality.*
- Three ways of guiding MDP policies:
  - Additional reward or penalty attached to states with a specific property (e.g., achievement of quality in an enabling method by a specified deadline).
  - "Nonlocal" proxy methods representing the committed actions of others (e.g., synchronized start times).
  - Hard constraints (e.g., using a release time to delay method starts until after an agreed-upon enablement).
- Hard constraints can be subsumed by nonlocal proxy methods.

Informed MDP Unrolling Performance

- Over a number of example problems, including GITI-supplied problems, the informed unroller is able to formulate policies with expected quality approaching the optimal, but a couple of orders-of-magnitude faster.
- Example for local policies for agents in the test1 problem: