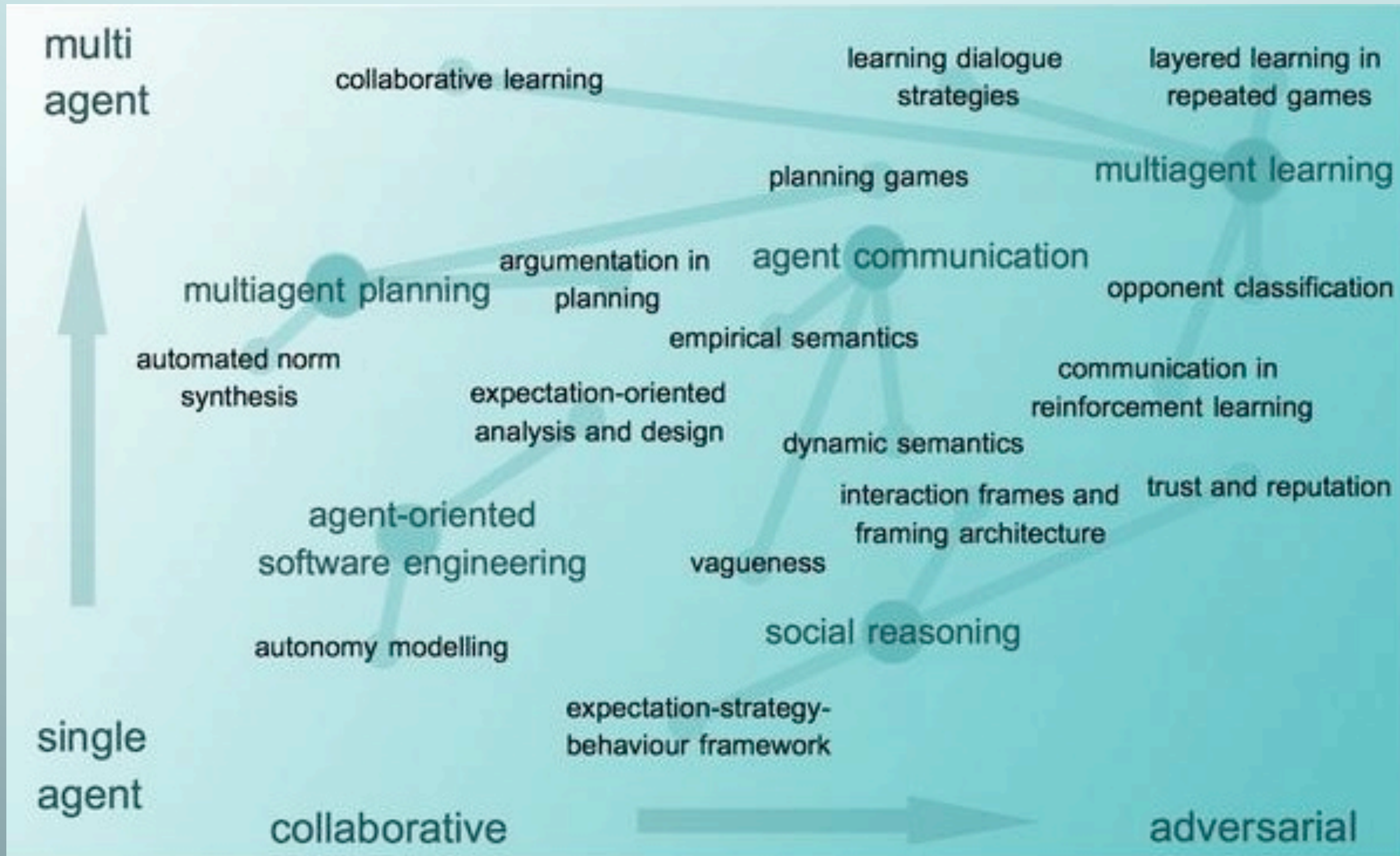
Reasoning about Interaction

Current Research at the Agents Group in Edinburgh

Michael Rovatsos
(joint work with Alexandros Belesiotis,
George Christelis, Matt Crosby, and Iain Wallace)

Centre of Intelligent Systems and their Applications
School of Informatics, University of Edinburgh
mrovatso@inf.ed.ac.uk

Background: Work in my group



Visit www.cisa.inf.ed.ac.uk/agents for details

Some motivation

- What is special about agents? Interaction in a common environment
- To make agents intelligent and autonomous, we need to automate such interaction
- Interested in knowledge-based reasoning about interaction
- Reasoning about interaction is by definition practical reasoning
- Vision: given a specification of the interaction problem, automatically synthesise behaviour

Practical reasoning about interaction

- We are interested in building systems, not only specifying them formally
- Rational agents need to synthesise action sequences to operate autonomously
- We want to tell them what to achieve, not how, abstraction desirable
- This suggests using knowledge represen-tation techniques
- Planning is the interface between KR methods and practical reasoning

Why not game theory?

- Game-theoretic methods very popular currently and address the problem of reasoning about interaction
- Information in real-world domains available in relational terms (e.g. on the Web), not enumerated state actions as assumed in game theory
- Non-incremental: unable to express how a game changes when we incrementally change background knowledge
- Knowledge-based methods might be useful in lifting overly restrictive assumptions (full rationality, perfect knowledge, etc)
- Intuition: many large-scale games might be actually "easier" than we think *(this is speculative)*

Current work

- Three examples of our current work in this area:
  - Macro-level: Automated norm synthesis
  - Meso-level: Argumentation-based conflict resolution
  - Micro-level: Practical social reasoning architectures
- Address general multiagent systems problems :
  - Setting up social laws to avoid undesirable states
  - Exchanging information to align divergent views
  - Reasoning about others from an agent’s point of view
- From a general computer science point of view:
  - Designer-level specification of system constraints
  - Integration of distributed sources of data
  - Process-level view of environment behaviour

Automated norm synthesis in a planning environment

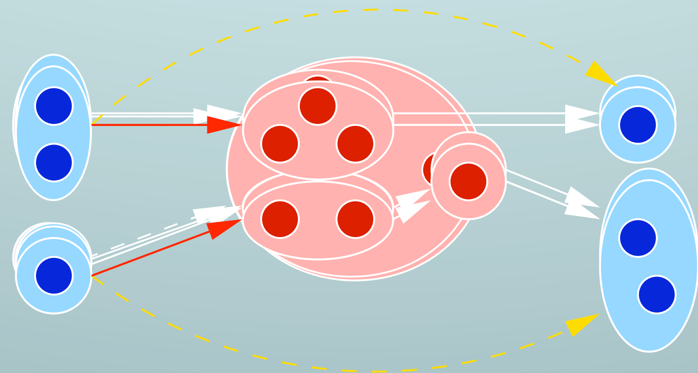
- Norms ensure global conflict states are never entered by prohibiting actions in certain states
- At the same time agents' private goals should remain achievable
- Automated synthesis of such norms is NP-hard in enumerated state systems
- Existing methods don't exploit abstractions of propositional/first-order domain theories
- Our method: find "detours" around conflict states by local search in generalised state spaces

The norm synthesis problem

- Assume a system with states S and some set of conflict states S_c
- Agents execute actions from a set A that change the global state
- Norm synthesis problem: compute a set of prohibitions (s,a) such that
  - S_c is never entered
  - any state in S that was reachable before is still reachable
  - not assuming specific initial states or knowledge about goals for the agents
- We assume that the norms will be adhered to, but could also look at automated synthesis of sanctions
- Traditional methods operating on enumerated state/ action spaces result in large sets of prohibitions, and don't scale well, so we attempt a relational approach

Automated norm synthesis

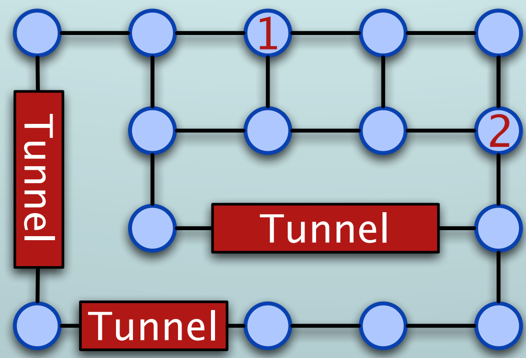
Iterated process of forward-backward search around conflict state specification:



- **Not better than full state-space search in the worst case but often get lucky**
- **With simple additional pruning techniques search can often be cut down drastically**
- **Currently working on synthesising sanctions**

Example

- **Tunnel world example:**



- **Agents entering tunnels have to leave them out the opposite end immediately (so on entering tunnel, future crash not avoidable)**
- **Our algorithm solves this by computing a general norm ({at_1(N), at_2(N'), tunnel(T), conn(N,T), conn(T,N')}, $move_1$(N,T))**
- **Note that we ignore extra cost caused to agent that has to take a detour to reach her goal when adhering to the norm**

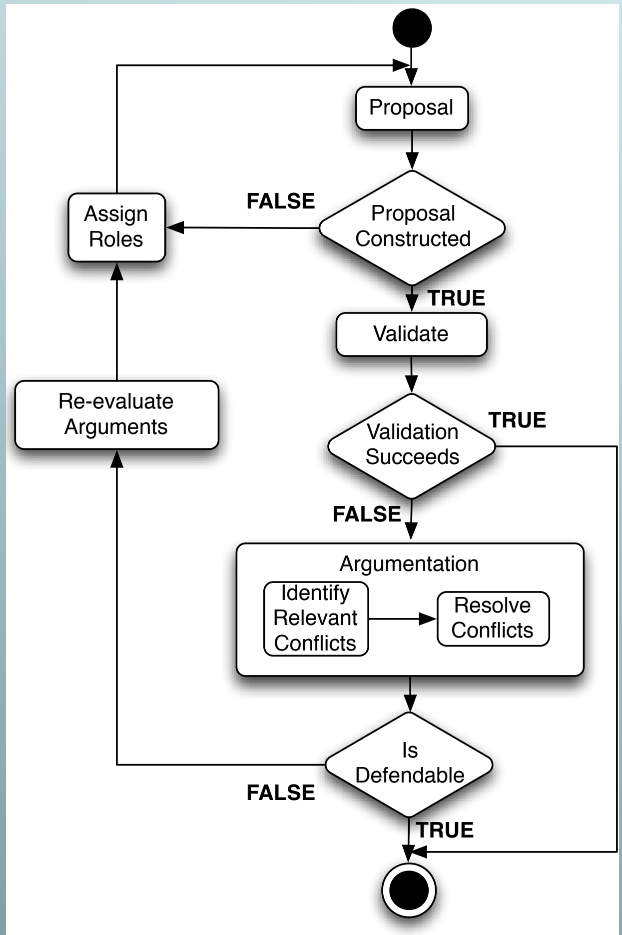
Argumentation-based conflict resolution in planning

- **Argumentation** is a method for determining the status of propositions in the presence of conflicting information
- Different acceptability-based semantics and protocols that implement these
- Rarely used for reasoning about action, our intuition is that this can be done more efficiently due to domain structure
- Suggest framework for **acceptable** planning:

  A plan P is acceptable wrt (potentially conflicting) knowledge bases KB_1 and KB_2

  iff $KB_1 \models P$ and $KB_2 \models P$

Argumentation-based conflict resolution



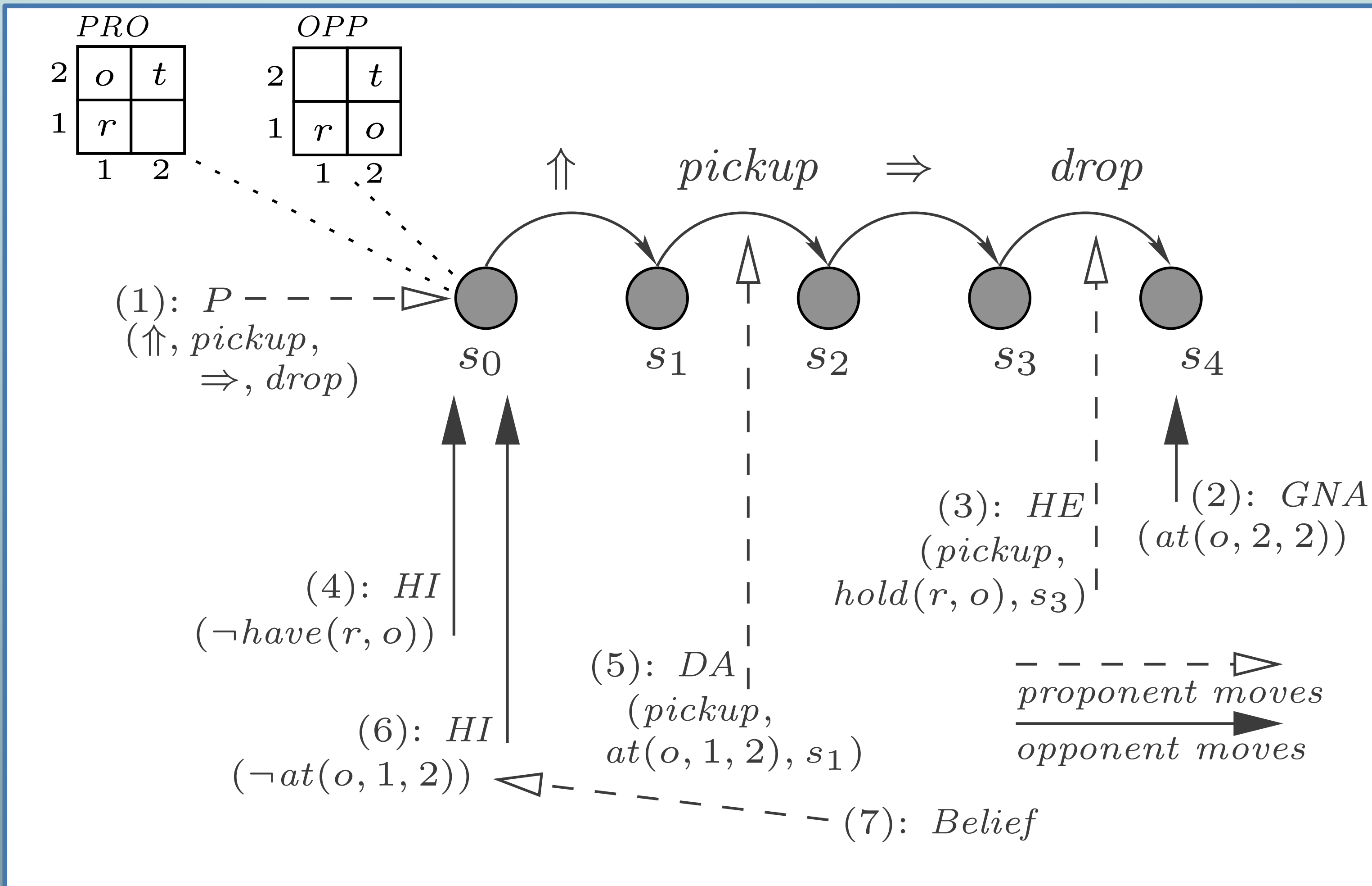
- Plan proposal generated by single agent (with any planner)
- Validation based on simple plan projection
- Dispute in case of disagree-ment, argumentation follows
- Ends in successful defence of initial proposal or rejection
- An alternative to generating one *P* that works under both *KBs*

Argumentation-based conflict resolution

- Planning domain represented in Situation Calculus
- Disagreement may exist regarding
  - initial state (including background knowledge)
  - planning operators (agreement on goal)
- Application of TPI-dispute protocol, but argument generation guided by plan structure
- Currently trying to extend method by updating local planning knowledge
- Also trying to extend method to planning with a defeasible planning theory
- Open problem: how to efficiently find plans that are possible using the combined knowledge of agents

Example

- **Robot gridworld domain**



Practical social reasoning architectures

- Practical reasoning architectures like BDI do not specifically consider social interaction
- Social reasoning = reasoning about other agents and social mechanisms governing the system (i.e. hidden system properties)
- Assumption:

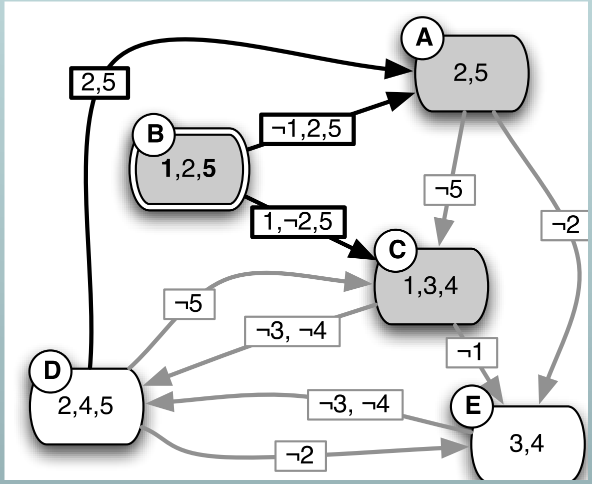
  any social reasoning mechanism can be formalised as a set of update rules regarding constraints concering hidden system properties
- Expectation-Strategy-Behaviour (ESB) architecture as a general computational framework

The ESB framework

- Expectations express assumptions about other agents' mental states or behaviours
- Their specification includes rules for how to update beliefs with relevant observations
- Strategies restrict the way potential future expectations are projected (think of a restricted expectation graph)
- Behaviours condition own behaviour (e.g. belief change at BDI level) on constraints verified against expectation graph
- Formal semantics, easily combined with state-of-the-art model-checkers
- An ESB engine can be easily combined with a normal BDI interpreter (in our implementation, Jason/ AgentSpeak

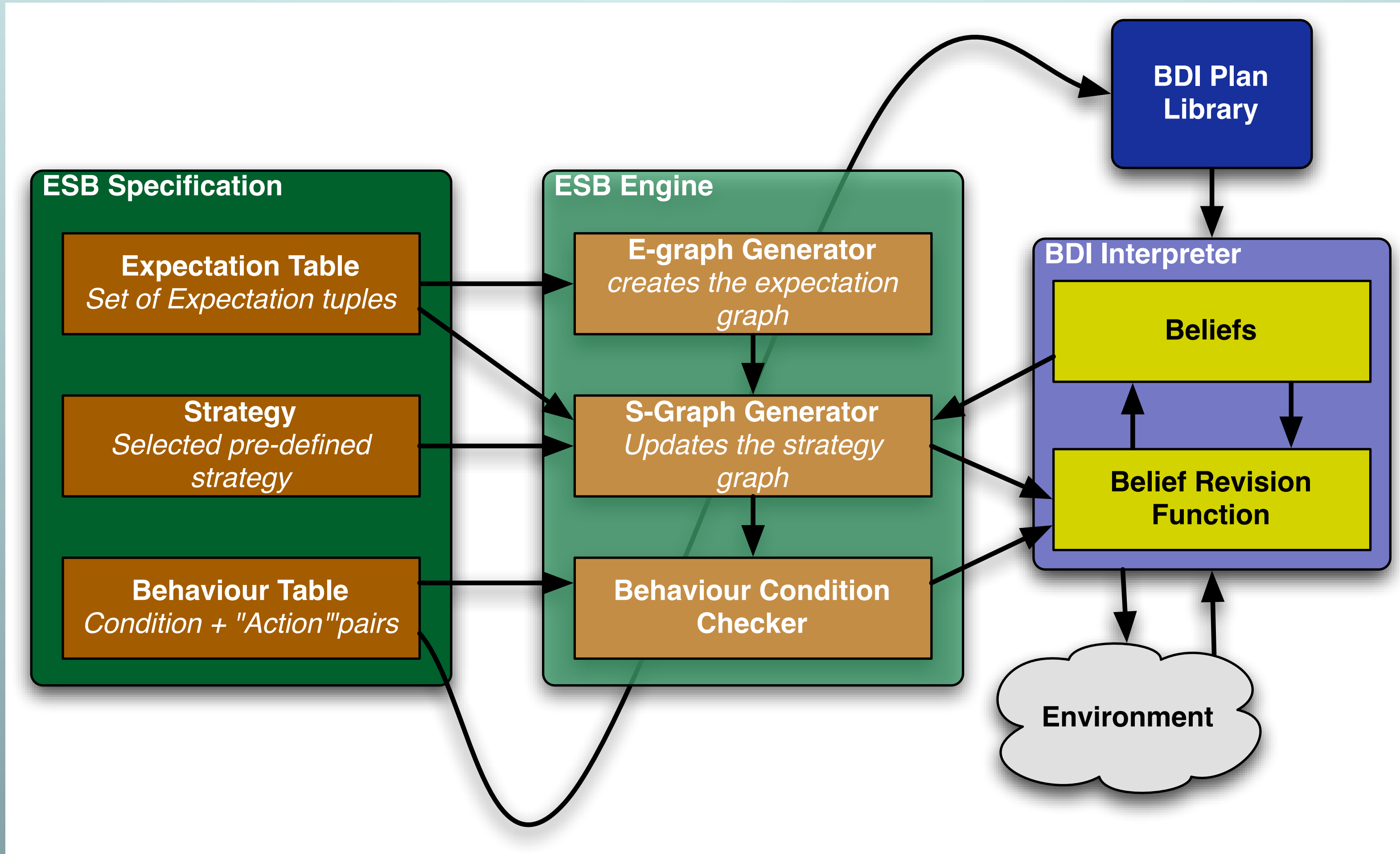
Reasoning in ESB

- Designer specifies expectations, strategies and behaviours in a declarative, modular way
- ESB engine constructs state transition system, restricted by strategy



- Model-checker verifies conditions on behaviour rules, and modifies BDI beliefs when behaviour rules fire

ESB reasoning engine



So what?

- Our current work addresses specific problems of reasoning about interaction
- But fragmented and very specific, would like solutions for more general problems
- Strongest contribution of agents to general AI is consideration of multiple (potentially conflicting) goals
- With “practical reasoning” glasses on, this suggests looking at strategic planning problems
- Very little work in this area, will discuss most recent approach

Brafman/Domshlak/Engel/ Tennenholtz (IJCAI 2009)

- Introduce notion of coalition-planning game (reward for goal, cost for plan, no action = 0)
- Solution stable if no set of agents can increase utility by jointly adopting other plan
- Formally: plan π *stable* for iff no plan π exists for any subset Φ' of agents Φ such that $u_\phi(\pi') > u_\phi(\pi)$ for all ϕ in Φ'
- Present an algorithm for computing stable plans, but complexity issues (enumeration of strategies necessary)

Interesting problems

Three general problems seem interesting:

- How to compute acceptable plan given a solution criterion (in particular adapting existing planning heuristics)
- How to search plan space incrementally for generating proposals during negotiation
- How to use background knowledge to guide plan recognition and optimal response generation

Evaluation

- No good benchmarks for MAP exist because research is fragmented
- Too many different potential problems to be accommodated
- Single-agent planning benchmarks can be adapted but is this useful?
- Multiagent systems people also interested a lot in continuous planning
- But performance metrics domain-dependent in this case

A good application?

- **Dialogue planning metaphor** covers synthesis, negotiation, and execution aspect
  - If communication actions are interpreted in a planning-based way, we should be able to plan them just like physical actions
  - But hard to decide about communication strategy before having synthesised collaborative plans
  - Actions planned for deception detection ahead of execution may affect suggested deals

Examples

BUYER-SELLER

B: I would like an art history book.

S: Good art history books range from $35-$55.

B: I would like something cheaper.

S: There's "Art for Kids" at $15.

B: I want a book for adults.

S: There's "Art History for Dummies" at $25.

B: Great, I'll take that.

(execution follows, including payment, delivery, etc)

PEER-TO-PEER

P: I'd like to stream a music concert in high quality tomorrow night.

Q: Who will be performing?

P: It's a "best-of" transmission from a festival.

Q: I don't like watching concerts unless I know what bands are playing.

P: Could I still borrow your bandwidth?

Q: OK, if you grant me prioritised access to yours for seven days after that.

(*execution follows, including settings to preference in P2P system, actual streaming actions, etc)*

Conclusions

- Reasoning about interaction crucial to multiagent systems
- Must involve planning one way or another, but no standard simple frameworks for multiagent case
- Some of our own work shows that planning formalisms are useful
- To develop more generic problems need convincing, simple examples
- Looking at multiple goals is (in my opinion) the strongest thing that multiagent perspective can add to single-agent planning
- Current solution concept proposals lead overly complex, more approximate methods needed

Thank you. Questions?

Material based on
Christelis & MR @ AAMAS 2009
Belesiotis, MR & Rahwan @ ArgMAS 2009
Wallace & MR @ AAMAS 2009

Find out more/get involved at
http://www.cisa.inf.ed.ac.uk/agents