G53DIA:
Designing Intelligent Agents

Lecture 6: Deliberative Architectures II

Brian Logan
School of Computer Science
bsl@cs.nott.ac.uk

Outline of this lecture

- the need for accurate world models
- approaches to representational error
- example: Quakebot
- approaches to deliberation time
- commitment strategies (deliberation about ends)
- example: Tileworld/IRMA

Recap: deliberative architectures

- in a deliberative architecture percepts (or communication) give rise to *goals*—representations of a state to be achieved
- the agent *deliberates* about how to achieve the goal
  - deliberation involves (usually systematic) *exploration of alternative courses* of action
  - a *deliberative architecture* typically includes automatic generation and comparison of alternatives
- result of deliberation is a *representation* of of the action(s) to be performed

Recap: the role of representations

- deliberation involves the manipulation of a *model of the world* and *possible courses of action*, rather than the world itself
- model of the world is a representation of the *current state* of the agent's environment
- to represent desired future states (goals) and courses of action (plans) some states of the agent must be *counterfactual*
- to generate counterfactual states, an agent must be able to represent actions and derive the consequences of actions *without actually performing them*

Recap: advantages of deliberation

- useful when the penalty for incorrect actions is high, e.g., when the environment is hazardous
- allows us to code a *general procedure* for finding a solution to a *class of problems*
  - may be easier to cover a wider range of task environments
  - may be better than reactive agents at coping with novel problems

Problems of deliberative architectures

- deliberation requires accurate world models
- hard to offer real-time guarantees on performance:
  - solving any given problem takes longer than an equivalent reactive implementation
  - solving different problems takes different amounts of time

Accurate world models

- agent must be able to:
  - update its model of the current environment based on its percepts
  - generate the counterfactual part of the model by correctly predicting how the world will change, e.g., as a result of its actions
- if the world model is incorrect, ‘correct’ deliberation may select the wrong course of action

Representing the real part

- it is difficult to create and maintain the veridical model of the current state of the world required for deliberation:
  - the information available to the agent may be *incomplete*, e.g., if the environment is only partially observable
  - the information available to the agent may be *inaccurate*, e.g., if the environment has changed since the representation was last updated or the agent's sensors give incorrect information

Representing the counterfactual part

- to generate hypothetical future states which are useful, the agent must be able to accurately predict how the world will change:
  - as a result of the *agent's actions*—agent's actions are often assumed to be infallible;
  - as a result of *exogenous changes*—actions of other agents or environmental changes are assumed to be predictable

Approaches to errors in representation

- ignore them and hope for the best, e.g., classical planning
- try harder to make the representation fit reality, e.g., by using more and better sensors, or by trying to ensure that actions really are infallible, e.g., by implementing them as robust reactive behaviours
- explicitly represent and reason about uncertainty, e.g., probabilistic representations, decision theoretic approaches
- interleave planning and acting–plan only a little way ahead and keep checking that things are going according to plan; if not, replan.

Example: Quakebot

- the Quakebot (Laird 2000) is an expert in playing Quake II death matches
- the perceptual information and motor commands available to the Quakebot are similar to those of a human playing the game
- to navigate, the Quakebot explores a level and builds up a map based on range data to walls

Example: Quakebot 2

- the Quakebot was implemented using the SOAR cognitive agent architecture
- original Quakebot did not do any planning
- used a fixed decision procedure to decide which action should be executed

- essentially a reactive agent (with state) which integrated large number of tactics
- however to improve the play of the Quakebot, more and more specialised tactics had to be added

Example: Quakebot 3

- later versions *anticipated* its opponent's actions by creating an internal representation of what the Quakebot thinks the opponent's internal state is, based on its observations of the opponent
- it then *predicts* the opponent's behaviour by using its own tactics to select what it would do if it were the opponent
- using simple rules to internally simulate external actions in the environment, the Quakebot forward projects until it gets a prediction that is useful, *or there is too much uncertainty about what the opponent would do next*
- prediction is used to set an ambush for the opponent or deny them a strategic advantage

Example: Quakebot 4

- although the Quakebot does not plan in conventional sense, it is still a deliberative agent
  - it can represent how the world might be in the future as distinct from how it is now
  - it can represent (sequences of) actions
  - it can derive the consequences of actions without actually performing them
- however it applies its predictive abilities to *other agents* rather than itself

Problems of deliberative architectures

- deliberation requires accurate world models
- hard to offer real-time guarantees on performance:
  - solving any given problem takes longer than an equivalent reactive implementation
  - solving different problems takes different amounts of time

Deliberation is inherently serial

- generation of each alternative is inherently serial, since each step relies on the state produced by the previous step
- the number of courses of action grows exponentially with the length of the solution, rather than linearly in the number of percept-action pairs
- the number of alternatives an agent can pursue in parallel is bounded by the number of processing units available
- consideration of some alternatives must be deferred, i.e., processed serially

Deliberation takes more time

- in a reactive architecture, if we know which action(s) to perform in a given situation we can perform them immediately
- all other things being equal, deliberation will take more steps than simply reacting, since we have to (serially) generate and compare alternatives
- if the agent doesn't learn, it must re-solve a problem each time it encounters it, and will *always* be slower than a reactive agent
- in a dynamic environment a deliberative agent may take too long to select an action, e.g., if the situation changes before deliberation is complete

Deliberation takes unpredictable time

- there is no one best problem-solving method ("No Free Lunch Theorem")
  - different problems are more or less difficult for different techniques
  - it is difficult to tell how hard a problem is just by looking at it–we have to try and solve it
  - we don't know how long it will take to find a solution to a problem in advance
- time required by a deliberative agent will vary from problem instance to problem instance

Approaches to deliberation time

- ignore it and hope for the best, e.g., classical planning
- make deliberation run faster, e.g., by using non-optimal algorithms which sacrifice solution quality for speed
- try to predict how long deliberation will take to solve the current problem, e.g, empirical AI
- adapt the amount of deliberation performed to the time available, e.g., anytime algorithms
- accept that deliberation may occasionally be too slow for some environments or some problems, e.g., monitor the environment and replan if the environment changes

Empirical AI

- difficult to tell how hard a problem is for a given problem-solving technique without trying to solve it
- *empirical AI* tries to characterise which problems are hard or easy for a given technique
- most of this work has been in the areas of optimisation, planning and constraint satisfaction problems
- concerned with which problems are soluble *at all*

Anytime algorithms

- interruptible algorithm which produces a *sequence of solutions* of *monotonically increasing quality*

- anytime algorithms can be used in two ways:

  – to compute the optimum amount of time to spend deliberating to maximise the utility of the outcome, e.g., decision theoretic approaches

  – deliberate for as long as possible before acting, on the assumption that deliberating for longer will produce a better solution

Planning in dynamic environments

- if the situation changes while the agent is planning or executing its plan, the agent should replan *if the changed situation invalidates the plan*

- it is difficult to make all the assumptions underlying the plan explicit– the 'qualification problem'—so most deliberative agents either:

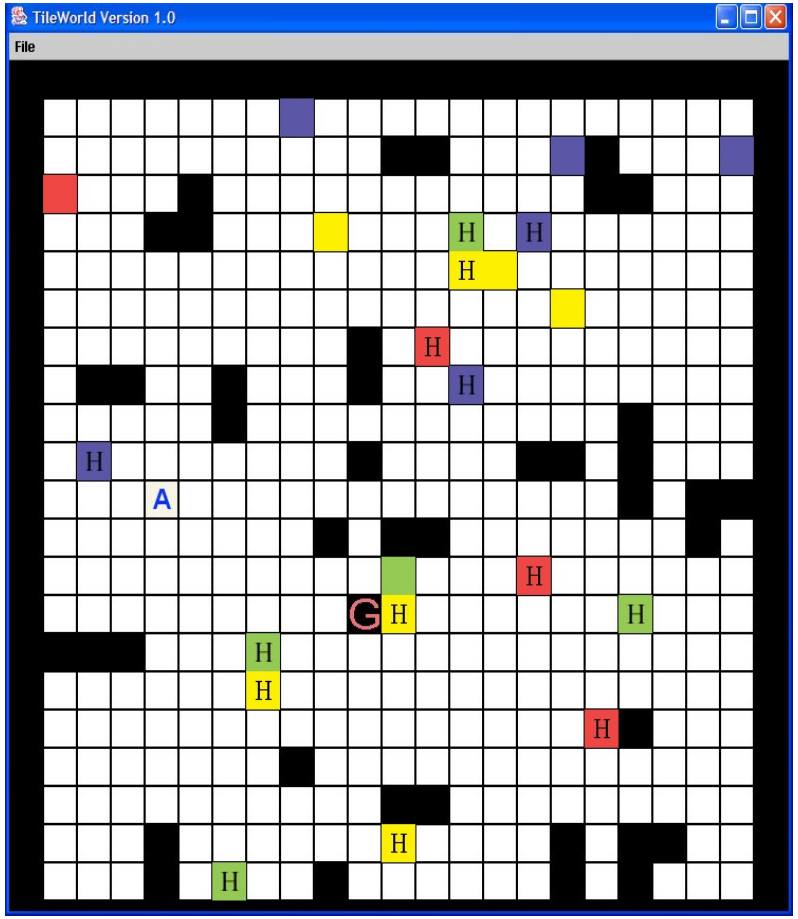
  – replan if there is *any* change–in case the change is significant

  – ignore *all* changes until one of the steps in the plan fails and triggers replanning

- *commitment strategy* determines when an agent will reconsider its current intentions

Example: the Tileworld/IRMA

- Tileworld is a testbed for agent architectures
- the environment consists of tiles, holes and obstacles
- the goal of the agent is to put the tiles in the holes to gain points
- the environment is *dynamic*—tiles, holes and obstacles can appear and disappear



Filling holes

- to gain points the agent must put tiles in the holes
- holes have different depths and the agent gets no points until the hole is filled
  - filling a deep hole will gain more points, but there is an increased chance that the hole (or the tiles the agent plans to use to fill it) may disappear before the hole is filled
- the agent gets most points for matching the colours of tiles and holes
- carrying more tiles uses more gas and the agent must periodically refuel at the gas station to avoid running out of gas

Tileworld agent task environment

- **environment:** observable, dynamic, non-deterministic, discrete, no other agents (in the original Tileworld)
- **actions:** infallible, have differing utilities and costs, agent is mobile, no communication with other agents
- **goals:** autonomously generated achievement and maintenance goals, weakly committed to its (achievement) goals, goals have differing utility and are episodic, may have meta-goals (IRMA)

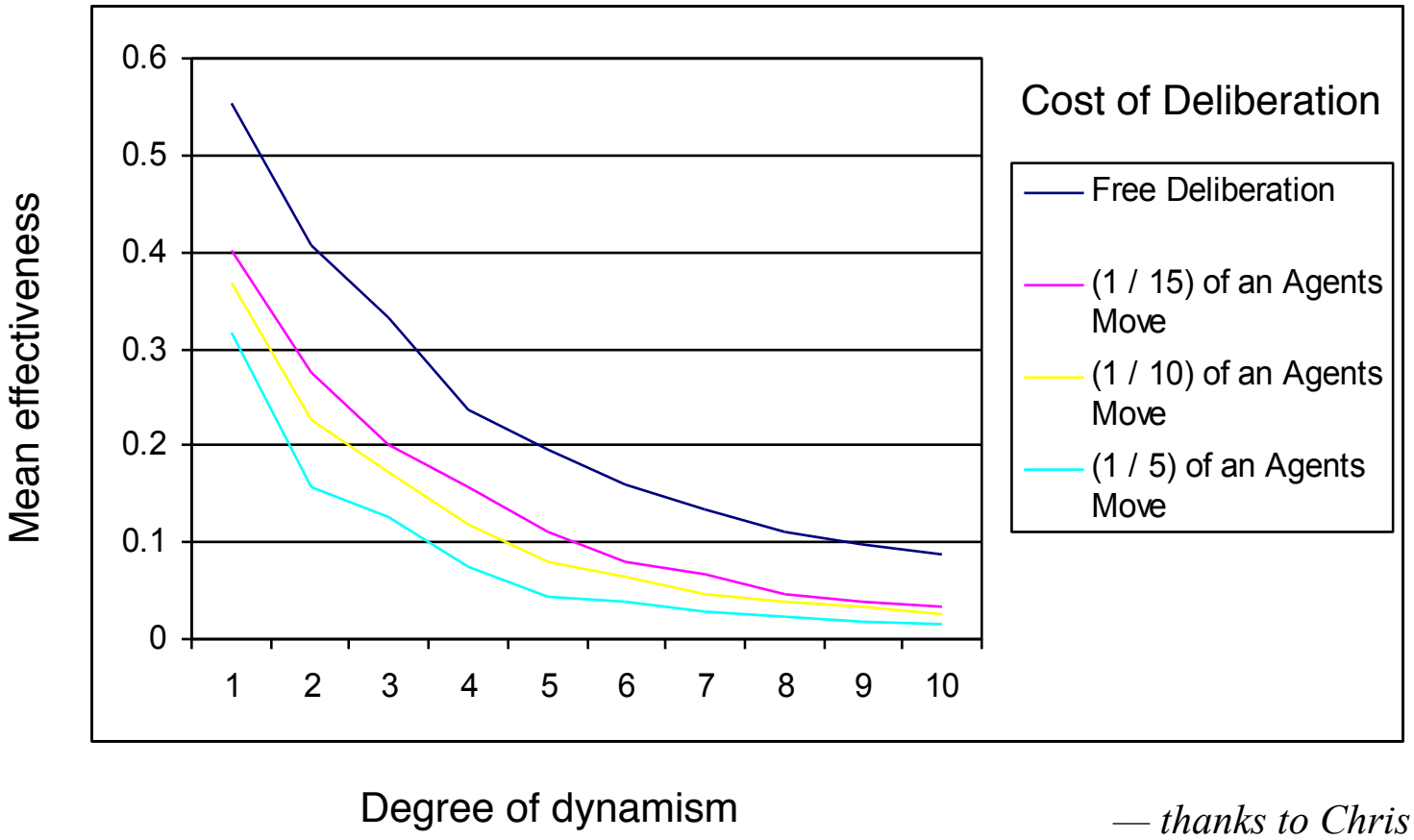
Tileworld agent goals

- the Tileworld agent has several different types of top-level goal (options):
  - wandering about
  - filling holes with tiles
  - building stockpiles of tiles of a given colour at particular strategic locations on the grid
  - getting gas

Simple Tileworld agent

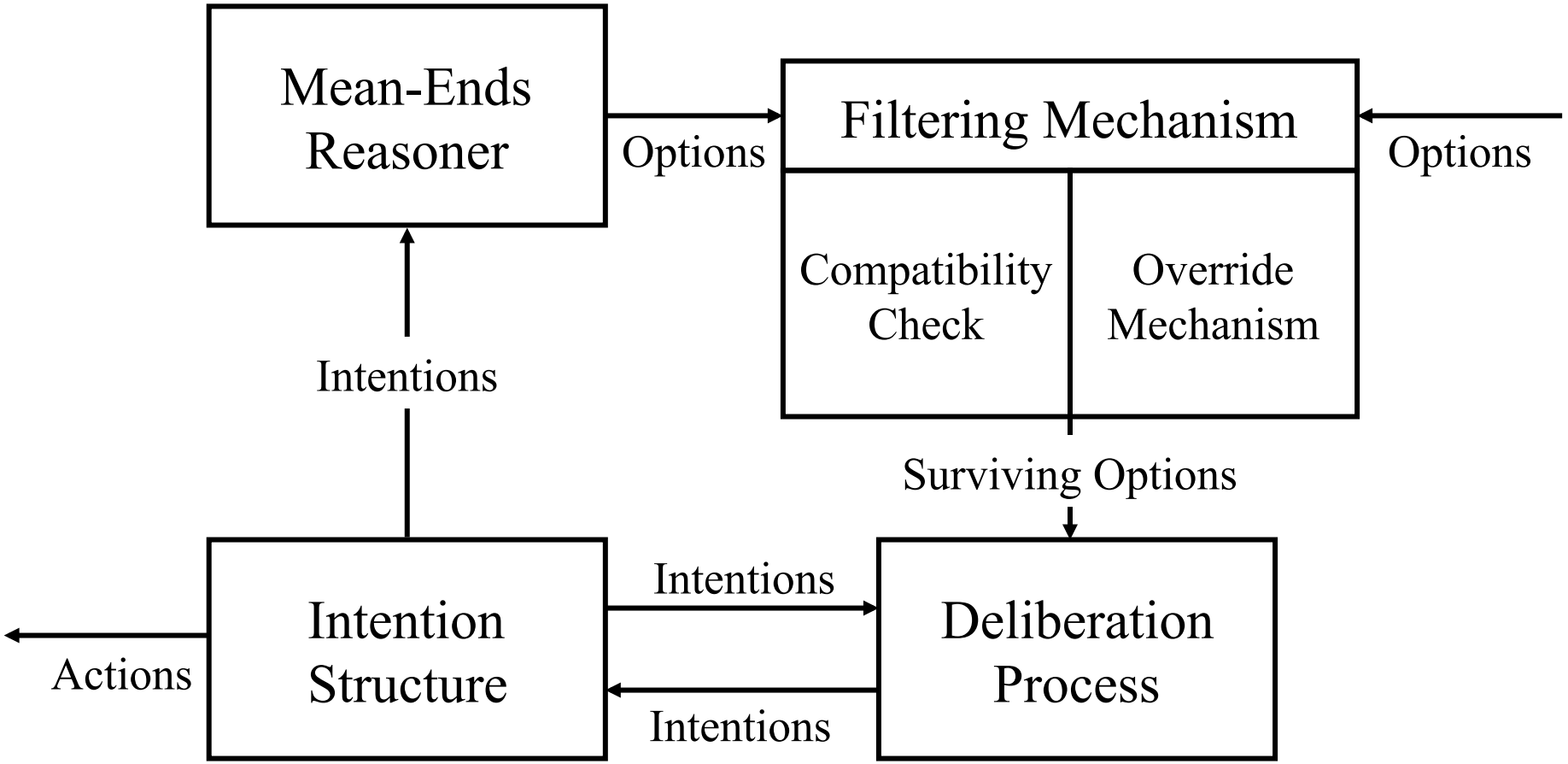
- the simple Tileworld agent generates a plan to fill the (current) holes in the Tileworld with (currently) available tiles
- whenever there is a new *option*, i.e., whenever a hole or tile appears or disappears, the simple Tileworld agent checks to see if it can produce a better plan than its current plan
- we can perform experiments to discover how the agent's success (score) varies with how long it takes to deliberate
- *deliberation cost* is expressed in terms of how many squares the agent could have moved if hadn't deliberated (agent doesn't move while deliberating)

The cost of deliberation



— *thanks to Chris Peel*

The IRMA architecture



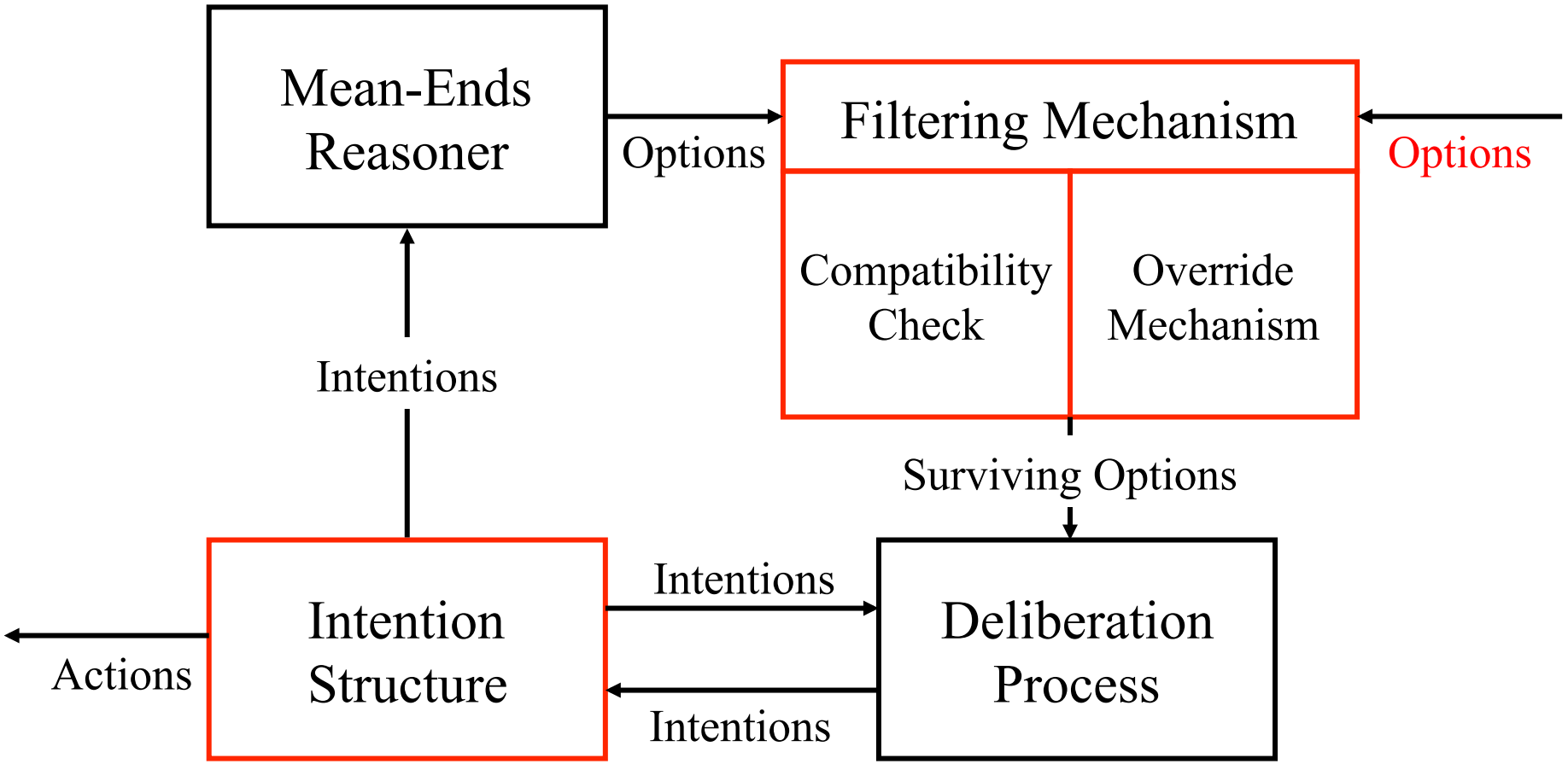
The IRMA processing cycle

- *perceiving* the world—the Tileworld is observable and the agent can determine whether there are any new opportunities for action (e.g., new holes to be filled)

- *reasoning* about which actions to perform
  - filtering potential options
  - deliberating about which of the remaining options to adopt as intentions
  - means-ends reasoning to produce options (plans) to achieve existing intentions

- *executing* the actions—acting involves performing a fixed number of primitive actions computed by means-ends reasoning in previous cycles

The IRMA architecture



Options and intentions

- while reasoning about which actions to perform, the agent maintains two lists:
  - a list of *options*—the actions the agent could perform but to which it is not yet committed
  - a list of *intentions*—the actions to which the agent is committed
- at any time, one intention may be singled out as the *current intention* —the action the agent is in the process of executing
- any new option for immediate action is assumed to be incompatible with the current intention (if there is one) and will be filtered from further deliberation unless it triggers a filter override

The role of filtering

- the IRMA architecture has a filter which restricts deliberation to options that are *compatible* with the agent's *already intended actions*

- for a new option to pass the filter, it must either pass:

  – the *compatibility check* which checks new options for compatibility with existing plans; or

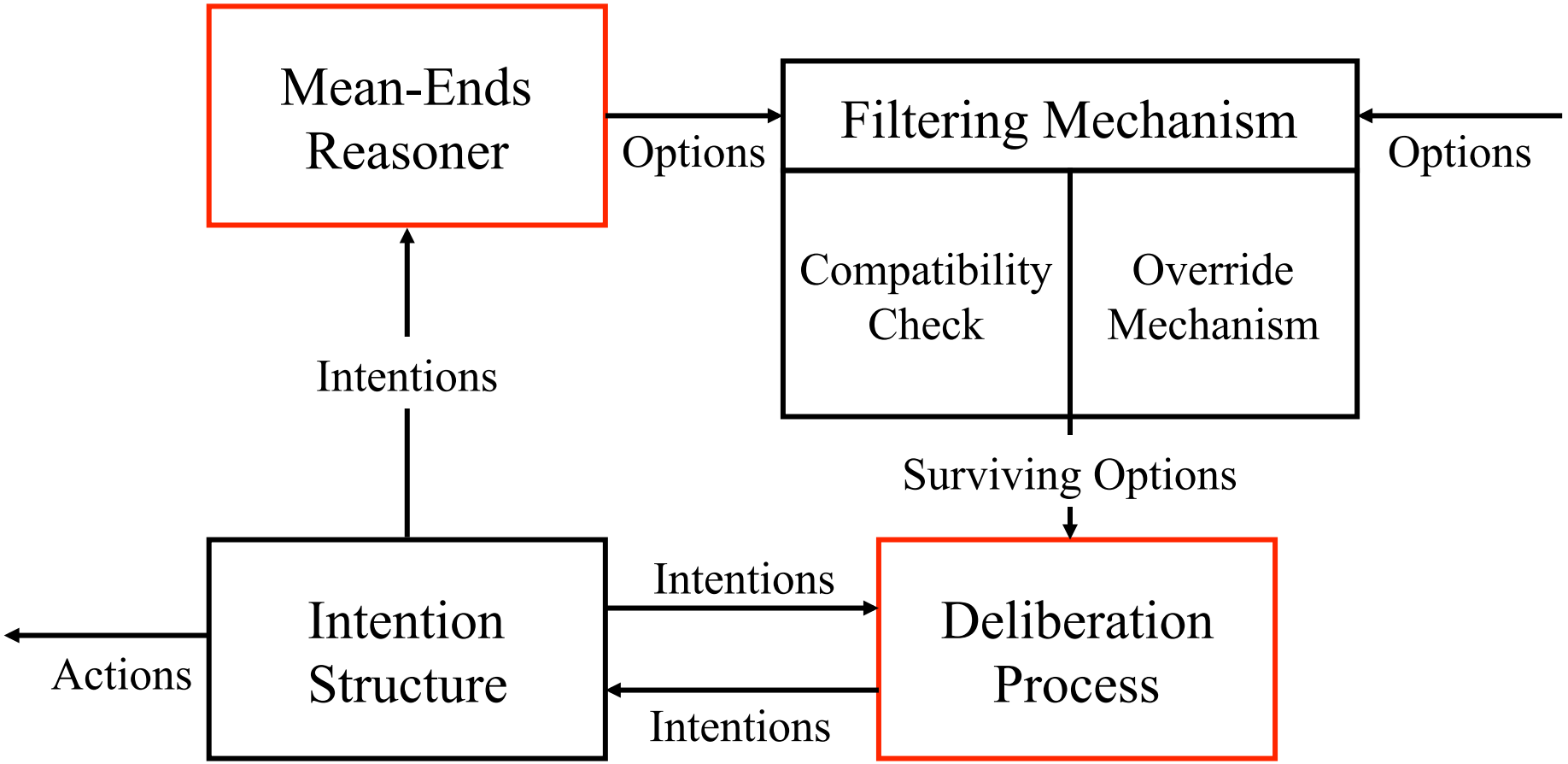
  – the *override mechanism* which encodes the conditions under which some portion of the existing plan(s) is to be suspended and weighed against a new option

- only options that pass the filter are subject to deliberation

The override mechanism

- an estimated value v_o is computed for each incompatible option o
  - v_o of a fill-hole option is the score that would be received if the hole is successfully filled with tiles of the correct colour
  - v_o of getting gas is a function of the current gas level
  - v_o of other options (e.g., stockpiling tiles and wandering) is a low constant
- for o to trigger an override and thus survive as an option for deliberation, v_o must exceed the computed value of the current intention v_c by at least the filter threshold t
- an agent with a negative threshold will deliberate about options that, *prima facie*, are less valuable than the intention with which they conflict

The IRMA architecture



Deliberation and Means-Ends reasoning

- in IRMA, *deliberation* and *means-ends* reasoning are two different processes:
  - deliberation involves deciding *which* of a given set of options to pursue, e.g., which hole to fill
  - means-ends reasoning involves determining *how* to achieve a given goal, e.g., which tiles to use to fill the hole
- means-ends reasoning produces additional options—ways to achieve a given goal—which can themselves be the subject of further deliberation

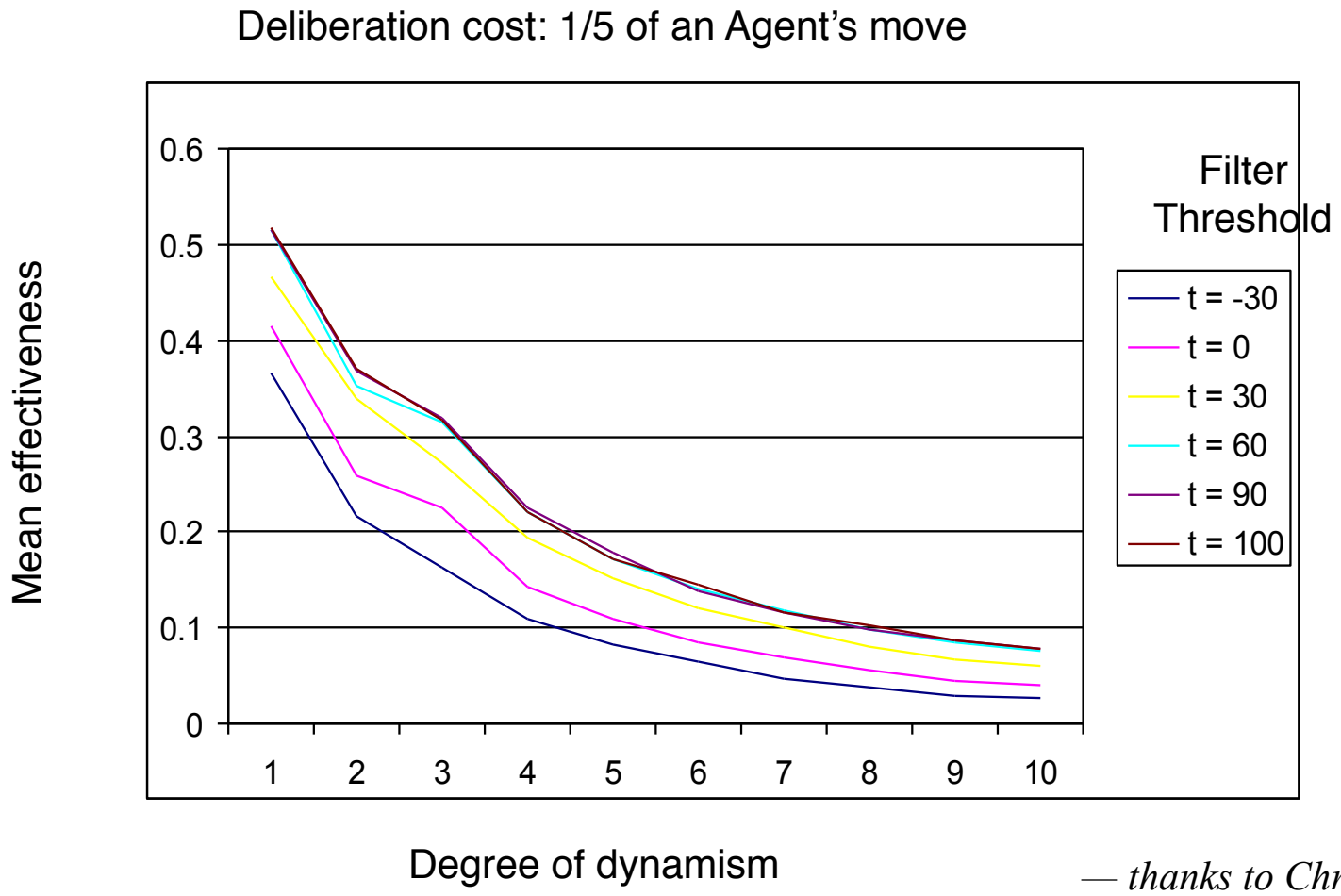
Deliberation

- deliberation weighs the alternative courses of action to determine those to which the agent should form commitments
- for example, when deliberating about a fill-hole option, the size and location of the hole, as well as its maximum score are considered
- if, as a result of deliberation, the highest-ranked option is determined to have a higher value than the current intention, it will replace the current intention in the intention list
- if there is no current intention, the highest-ranked option or intention becomes the new current intention

Means-Ends reasoning

- means-ends reasoning (path planning in the case of the Tileworld) occurs whenever there is a new current intention
- the Tileworld agent uses a special purpose planning algorithm which produces two plans
  - one for filling the hole with matching tiles
  - one for filling the hole with whichever tiles are closest
- further deliberation is then used to decide between these

The effect of filtering



— *thanks to Chris Peel*

The limits of filtering

- in a wide range of Tileworld environments, agents perform better when they filter from consideration actions which are incompatible with their existing intentions

- the agents which performed best reconsidered their current goal only if it became impossible (e.g., because the hole it was attempting to fill disappeared) or an emergency arose (e.g., the agent was about to run out of gas)

- agents with lower filter thresholds do better in environments with frequent low-payoff options and rare high-payoff options with tight deadlines.

— (Pollack et al 1994)

The next lecture

Hybrid Architectures

Suggested reading:

- Arkin (1998), chapter 6.