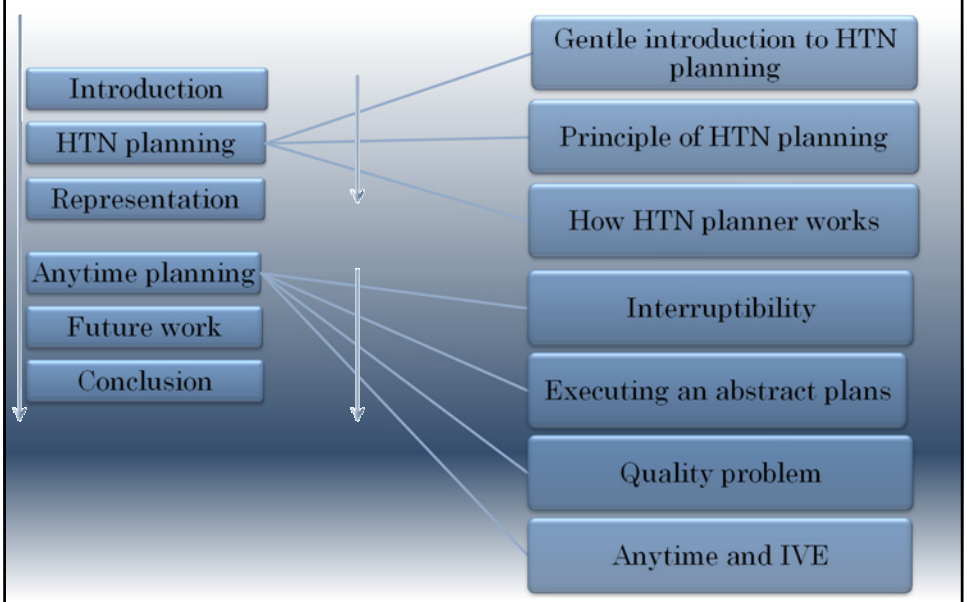
HTN planning for virtual humans

Ondřej Holeček

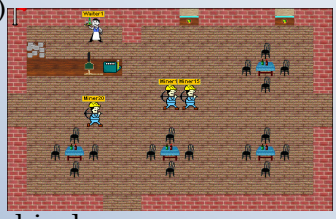
anzo@seznam.cz
http://aaannz.ic.cz

What I will talk about



IVE = Intelligent Virtual Environment

- ✓ Toolkit for discrete simulation of large number of autonomous agents
- ✓ Highly extensible
- ✓ Platform independent (written in Java)



IVE principle:

- Simulating effectiveness ensured by Level-Of-Detail technique
- Theory of affordances
- Basic Genius is driven by a set of hierarchical reactive if-do-then rules
- Genius can change the world by executing processes

IVE website: http://urtax.ms.mff.cuni.cz/ive

Current situation:

Genius is controlled by reactive planning (set of if-do-then rules)
Every time genius wants to make decision if-do-then rules are evaluated
Plan means only which action(process) to execute next
Genius can`t look forward to analyze possible solutions for his goal
Solving a little bit more complex task may be an overwhelming obstacle

Desired situation:

Genius is controlled by external (selectable) planner
Plan is generated only on some special events
Generated plan is supposed to solve whole problem
Genius is capable of solving complex task

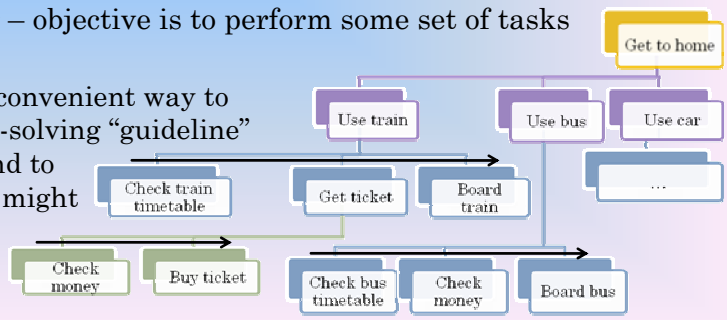
Gentle introduction to HTN planning

HTN ~ Hierarchical Task-Network

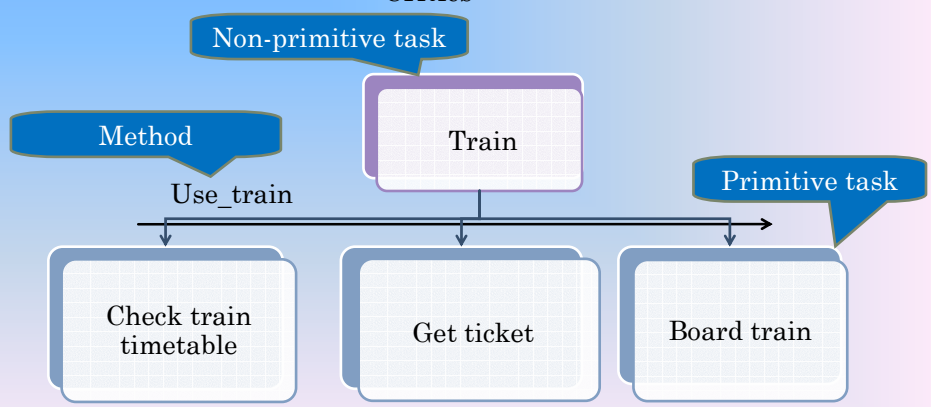
"HTN planners differ from classical planners in what they plan for and how they plan for it"
(Automated Planning : Theory and Practice)

Classical planner – objective is to achieve the set of goals
HTN planner – objective is to perform some set of tasks

HTN provide convenient way to write problem-solving "guideline" that correspond to how a human might think about solving the problem



HTN components:
- Tasks – Primitive tasks ~ action
  Non-primitive tasks ~ compound tasks
- Methods ~ task reduction schemas
- Operators
- Critics



The HTN Planning Procedure

1. Input a planning problem P
2. If P contains only primitive tasks, then resolve conflicts and return the result. If conflicts cannot be resolved, return failure
3. Nondeterministically choose task *t* in P
4. Nondeterministically choose method *m* to reduce *t*
5. Replace *t* with the task-network produced by applying *m*
6. Use critics to find the interactions among the tasks in P, and suggest ways to handle them.
7. Apply one of the ways suggested in step 6
8. Go to step 2

Problems with HTN planning : Require "hints" (methods)

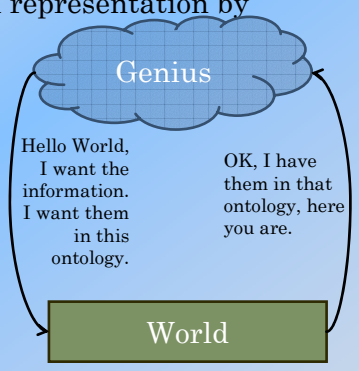
In this project, SHOP2 planner is used.
SHOP2 (Simple Hierarchical Object Planner 2) is Total-Order HTN planner

Representation for Basic Genius – native IVE
Representation for Planning Genius – extended PDDL

How to represent two world representation and maintain compatibility?

IVE supports multiple different world representation by using an ontology,
but it requires the world designer to write the world two or more times.

- one in IVE native format
- one in the requested ontology



Idea of automatic conversion

PDDL→IVE automatic conversion is necessary for executing generated plan.

Anytime planning

Classical planning problems are based on restricted world.
- world is static
- enough time for computing the plan

In dynamic world everything can change in every second.
There are usually more actors within one world.

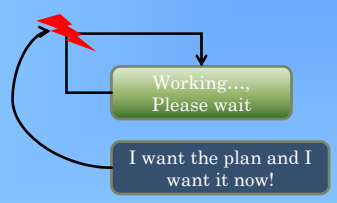
This raises new demands on planner subsystem.

First, we demand planner to be **interruptible** as we do not have so much time as in static world.
Second, after we interrupt planner, we demand planner to return **useful** plan.

Solution: **Anytime planning** system [Zilberstein, 1996]

Interruptibility

Problem is the answer returned after the interruption.



Different planning approaches produce different outcome.

The plan quality must not decrease with time spent by computing the plan

HTN planner works in cycles reducing an abstract tasks until it is expressed completely in primitive operators.
After every reduction, the resulting task network is an entire plan at varying levels of abstraction.

→ How to execute abstract plan?

Executing an abstract plan

There is no elegant answer to this problem, yet

Possible solutions:
- After-interruption abstract tasks refining
- Rely on library of reactive plans

After-interruption abstract tasks refining → in HTN there is usually more than one refinement

Library of reactive plans → for which actions and how many reactive plans?

In dynamic environment – basic need is a library of reactive actions and reactive plans

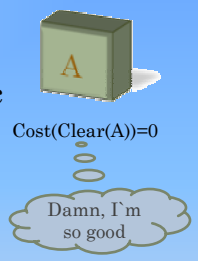
Quality problem

What is quality of plan?

- number of steps
- use of resources
- time necessary to execute plan
- …

} domain specific

- how abstract the plan is } our choice



The more abstract plan the costlier plan is. Why?

Because more abstract plan is less likely to be executed properly after an interruption.

How to measure "abstractness" of plan?

Count the number of refinement steps it will take to transform task to primitive.

Anytime in IVE

Implementation of an Anytime interface over IVE planning genius.

Based on A-UCMP planner – Anytime extension of UCMP planner developed by Nick Hawes

My goal – Generic anytime interface to any(almost) HTN planner.

- Plan is usable after interruption - idea based on using Waypoints
  - waypoint represent a subgoal on high level of abstraction
  - the point, that we most probably have to go through in order to achieve goal.
- Plan quality is monotonic with respect to time
  - planning-time heuristics
  - inspired by Hawes → HSP(Heuristic Search Planner

Future Work

Agent cooperation – delegated planning

- presence of some master who plans actions for workers
- or tactical intelligence common to agents

Speculative planning

- planner is able to cheat (assume some tasks that they are possible even thought the agent does not know how to perform it yet

Conclusion

- IVE
  - extensible framework for discrete simulation
  - what is genius and how is he going to change
- HTN planning
  - what it is
  - principle
  - pros & cons
- Anytime planning
  - principle
  - problems
  - implementation