PLANNING (SEARCH) IN GAMES
The Real and Virtual Worlds and a Not-so-optimistic Account on Planning

Martin Černý      cerny.m@gmail.com
OUTLINE

- Planning in the real world
  - What is NOT so simple
- Using planners
  - What I learned using planners in practice
- Planning as A*
- HTN
- Adversarial search

http://aigamedev.com/open/review/planning-in-games/
Planning in the real world
What is not simple

- Multiple problems:
  - Continuous worlds
  - Dynamicity
  - Non-determinism
  - Unknown domains
  - Partial observability
  - Goals
  - Reasoning about time
  - Speed of planning
  - Meta reasoning
We are still in PSPACE!

IPC – limit 30min

Games can offer only fractions of seconds

Contemporary STRIPS planners can handle approximately hundreds of predicates and/or action within a second (using a whole core)

Anytime planning
  • But how do you do that?
PLANNING IN THE REAL WORLD

METAREASONING

- When to plan/replan
  - Opportunism
- How much time do I have?
- When to stop planning?
  - Optimal plans are usually unnecessary
- Commitment
- Requires tight integration of the planner
PLANNING IN GAMES
WHAT DO THEY USE INSTEAD?

- Reactive techniques still prevalent
  - FSM
  - Hierarchical FSMs
  - Behaviour trees
- At some level, reactive techniques are indispensable
USING PLANNERS

- Modelling, modelling, modelling
- There are bugs.
  - Typically caused by too simple or “nonsensical” problems
  - Not so many (compared to other academical SW)
- Using planners from Java: Planning4J
  - Universal API for IPC planners nad JSHOP2
Recent IPC are quite opposed to realtime planning
- 30 minute timeout
- Fast Downward

Issues with PDDL:
- Except for BlackBox all planners I ever used (and a majority of planners in general) translate PDDL to state-variables (and it takes time)

No possibility to alter the course of planning, interrupt prematurely etc.
PLANNING AS A*

- Easy to understand
- State-of-the art IS heuristic forward search
- Procedural effects, procedural preconditions
  - Gravity, shooting,….
GOAP

- Goal oriented action planning
- The only documented system in games that uses the word “planning”
  - Used in other games as well
    - S.T.A.L.K.E.R.: Shadow of Chernobyl
    - Fallout 3
    - Deus Ex
    - ...
- Positive reception by players
- STRIPS-based
- [http://web.media.mit.edu/~jorkin/goap.html](http://web.media.mit.edu/~jorkin/goap.html)
The world is represented by state variables (a bit like CSP formalism)
- Variables may point directly to in-game entities

Procedural preconditions and effects

Action costs

A*

Separate system for goal selection
GOAP

Advantages and Disadvantages

- **Advantages**
  - Smarter AI
  - Separation of actions and goals from the implementation
  - Easier to maintain than FSM

- **Disadvantages**
  - No direct control over agent behaviour – difficult debugging
  - Many details to tweak

- **Reportedly, HTN and behaviour trees are favored now over GOAP**
**The Sims IV**

- Sims interact with objects
  - Autonomously
  - User-directed
- We do not want to stop interactions when not necessary
THE SIMS IV

- Let’s skip to Peter Ingebritson’s slides from GDC (slide 116)
BUILD-ORDER PLANNING

- Starcraft – BroodWar
- Depth-first branch & bound (no PDDL model!)
  - Saves memory in comparison to A*
  - Depth-first is FAST
- https://www.skatgame.net/mburo/ps/aiide11-bo.pdf
Hierarchical task networks (HTN)

- **Attack**
  - Flank
    - `Is_flankable(from, flankpos)`
    - `Run(from, flank_pos)`
    - `Shoot(flank_pos, to)`
  - **Attack directly**
    - `Shoot(from, to)`
    - `Run(from, to)`
Suppose: \texttt{Is\_flankable(mala\_strana, kampa)}

- \texttt{Attack(stare\_mesto, mala\_strana)}
  - \texttt{Flank}
  - \texttt{Is\_flankable(mala\_strana, kampa)}
    - \texttt{Run(stare\_mesto, kampa)}
      - \texttt{Attack(kampa, mala\_strana)}
      - \texttt{Attack directly}
  - \texttt{Flank}
  - \texttt{Is\_flankable(kampa, ???)}
    - \texttt{FAIL}
  - \texttt{Attack directly}

\texttt{Shoot(kampa, mala\_strana)}

\texttt{Run(kampa, mala\_strana)}
Variables + bindings…
Undecidable in the worst case!
May be considered an extension of POP
Better at modeling parallel actions
  • Often employed in team planning
Good at capturing domain-dependent knowledge
Tasks that are “puzzle-like” (e.g. Sudoku) are hard to express
HTN – APPLICATIONS

- Killzone 2
  - [http://www.slideshare.net/guerrillagames/killzone-2-multiplayer-bots](http://www.slideshare.net/guerrillagames/killzone-2-multiplayer-bots)
- Dark Souls, Max Payne 3, …
- Non-industry
  - Teams of bots in UT
  - Winner of the AIGameDev Capture the flag tournament
Adversarial Search

- Planning is just not enough!
- Rock-paper-scissors mechanics.
ADVERSARIAL SEARCH IN STARCRAFT

- Simplified combat model
- AlphaBeta considering durations
- Portfolio search
  - [https://www.skatgame.net/mburo/ps/combat13.pdf](https://www.skatgame.net/mburo/ps/combat13.pdf)
- Hierarchical adversarial search
MONTE-CARLO TREE SEARCH

- Highly recommended read
  - [http://en.wikipedia.org/wiki/Monte_Carlo_tree_search](http://en.wikipedia.org/wiki/Monte_Carlo_tree_search)

- Tested in RTS (Wargus) combat
    - But outperformed by the approaches on previous slide

- Used in Total War: Rome II (strategic level), Fable Legends (real time)
OUR WORK IN KINGDOM COME: DELIVERANCE

- Situations – find suitable NPC combinations with CSP
  - In 0.1ms…
- Monte Carlo Tree Search for Combat
SUMMARY

- Planning is good, BUT…
- IPC results do not correspond to performance in real time
- There are other techniques than STRIPS and they have their strengths
- In games you absolutely need procedural effects
- Contact me
  - cerny.m@gmail.com
  - Theses, projects: http://pogamut.cuni.cz/prace