Stavros Vassos, University of Athens, Greece stavrosv@di.uoa.gr May 2012

INTRODUCTION TO AI STRIPS PLANNING

.. and Applications to Video-games!

Course overview

- Lecture 1: Game-inspired competitions for AI research, AI decision making for non-player characters in games
- Lecture 2: STRIPS planning, state-space search
- Lecture 3: Planning Domain Definition Language (PDDL), using an award winning planner to solve Sokoban
- Lecture 4: Planning graphs, domain independent heuristics for STRIPS planning
- Lecture 5: Employing STRIPS planning in games: SimpleFPS, iThinkUnity3D, SmartWorkersRTS
- Lecture 6: Planning beyond STRIPS

Course overview

- Lecture 1: Game-inspired competitions for AI research, AI decision making for non-player characters in games
- Lecture 2: STRIPS planning, state-space search
- Lecture 3: Planning Domain Definition Language (PDDL), using an award winning planner to solve Sokoban
- Lecture 4: Planning graphs, domain independent heuristics for STRIPS planning
- Lecture 5: Employing STRIPS planning in games: **SimpleFPS**, iThinkUnity3D, SmartWorkersRTS
- Lecture 6: Planning beyond STRIPS

SimpleFPS planning benchmark

- Focus on first-person shooter (FPS) games and the non-player characters (NPCs) that act against the human player.
- Focus on goal-oriented action planning (GOAP) for NPC behavior.

SimpleFPS planning benchmark

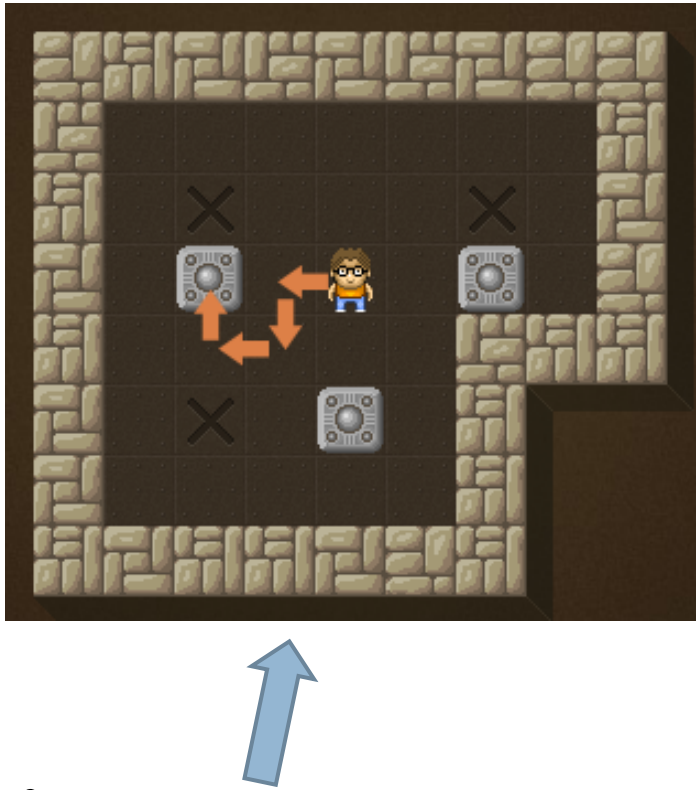
- Focus on first-person shooter (FPS) games and the non-player characters (NPCs) that act against the human player.
- Focus on goal-oriented action planning (GOAP) for NPC behavior.
- SimpleFPS is a first step towards evaluating how the existing academic approaches for planning would perform if directly applied in an FPS setting.

SimpleFPS planning benchmark

- Focus on first-person shooter (FPS) games and the non-player characters (NPCs) that act against the human player.
- Focus on goal-oriented action planning (GOAP) for NPC behavior.
- SimpleFPS is a first step towards **evaluating** how the existing **academic** approaches for **planning** would perform if directly applied **in an FPS setting**.

STRIPS Planning

- Given:
  - Initial State
  - Goal
  - Available actions



- Find:
  - A **sequence of actions** that satisfy the goal
  - E.g.: [**Left, Down, Left, Up, …**]

Planning Domain Description Language

- Language for specifying STRIPS planning problems
- Formal syntax like a programming language

  - Initial State (:init …)
  - Goal (:goal …)
  - Actions

```
(:action name
        :parameters (?from ?to ?dir)
        :preconditions (…)
        :effects (…)
)
```

Planning Domain Description Language

- Language for specifying STRIPS planning problems
- Formal syntax like a programming language
  - Initial State (:init …)
  - Goal (:goal …)
  - Actions (:action name
    :parameters (?from ?to ?dir)
    :preconditions (…)
    :effects (…)
)
- Literals like Lisp, e.g., bot-at(area1) (bot-at area1)

Planning Domain Description Language

- Language for specifying STRIPS planning problems
- Formal syntax like a programming language

Planning Domain

- Predicates (:predicates …)
- Actions

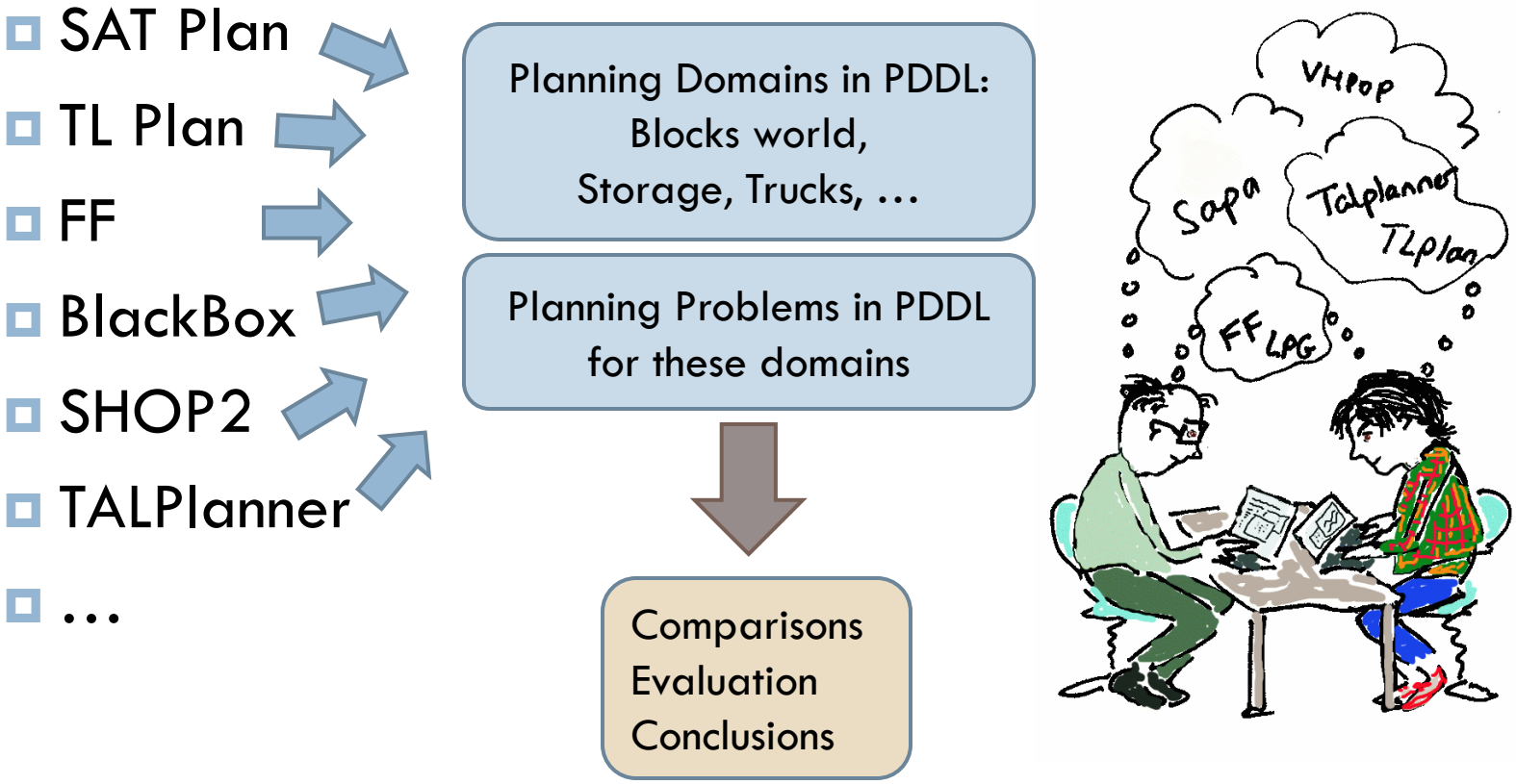
```
(:action name
        :parameters (?from ?to ?dir)
        :preconditions (…)
        :effects (…)
)
```

Planning Problem

- Objects (:objects …)
- Initial State (:init …)
- Goal (:goal …)

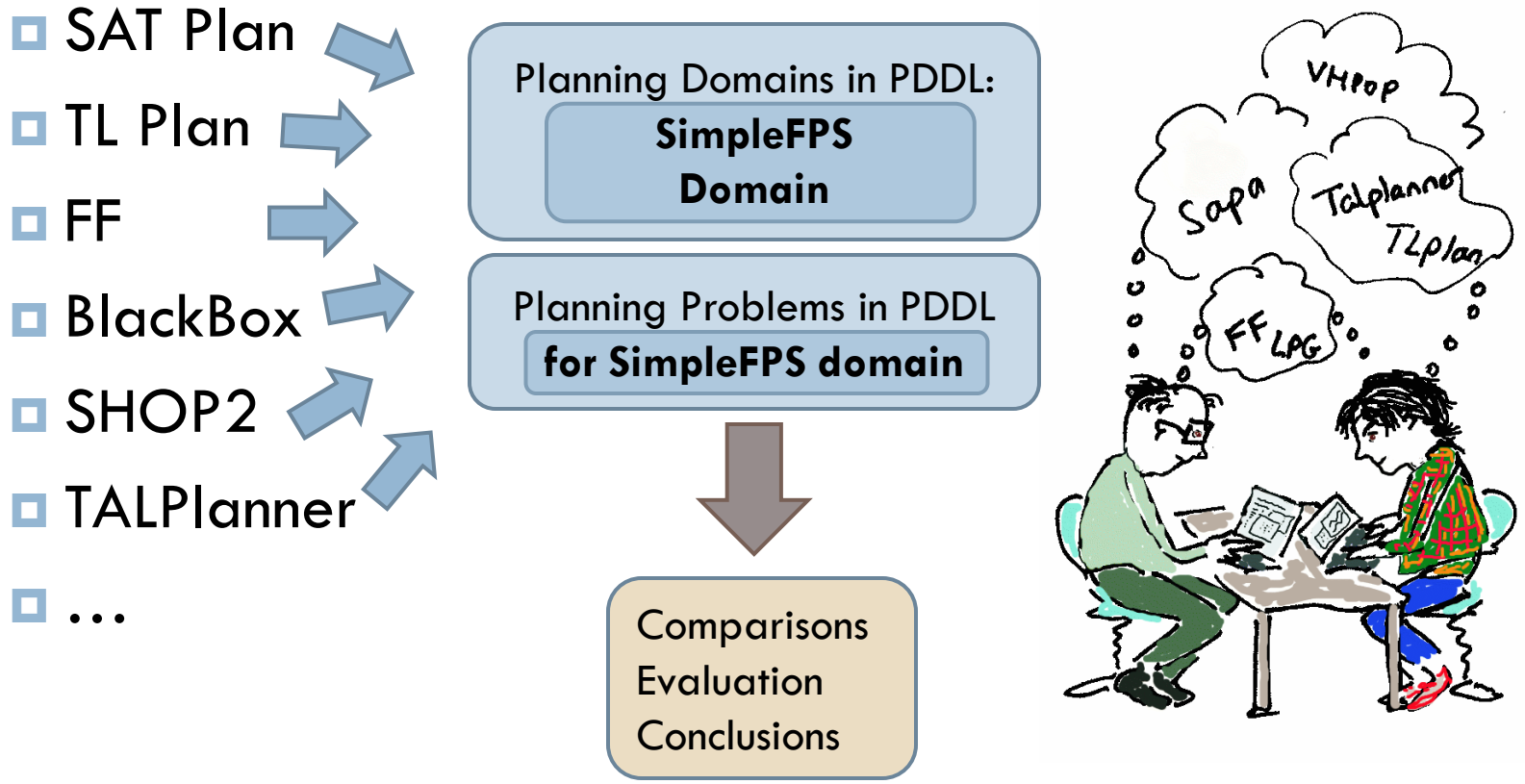
Planning Domain Description Language

- Planning Domain Description Language



Planning Domain Description Language

- Planning Domain Description Language



Motivation for SimpleFPS

- Planning in academia: extensively tested
- Planning in FPS video games: not extensively tested
- SimpleFPS: A PDDL domain for evaluating academic planning techniques for NPCs in First-Person Shooters

Motivation for SimpleFPS

- Planning in academia: extensively tested
  - Many PDDL planning domains and problems available
  - Many off-the-shelf PDDL planners available
- Planning in FPS video games: not extensively tested
  - A few success stories but not clear if the same works under different assumptions or what is the best approach
- SimpleFPS: A PDDL domain for evaluating academic planning techniques for NPCs in First-Person Shooters

SimpleFPS planning benchmark

- SimpleFPS_PDDL_Domain.txt

- SimpleFPS_PDDL_ProblemGenerator.c

SimpleFPS planning benchmark

- SimpleFPS_PDDL_Domain.txt:
  - Specifies the **predicates** that can be used to describe the initial state of the game-world and the goal condition for the NPC.
  - Specifies a list of **available actions** that the NPC can perform, along with their **preconditions** and **effects** in terms of the predicates of the domain.

- SimpleFPS_PDDL_ProblemGenerator.c:
  - A tool to generate problem instances, i.e., an **initial state** of the game-world and a **goal** for the NPC.

SimpleFPS planning benchmark

- Summer student project at the University of Athens
  - Michail Papakonstantinou
- AIIDE-2011 workshop paper
  - The SimpleFPS Planning Domain:
    A PDDL Benchmark for Proactive NPCs
- Code/datasets available online
  - http://code.google.com/p/simple-fps-pddl/

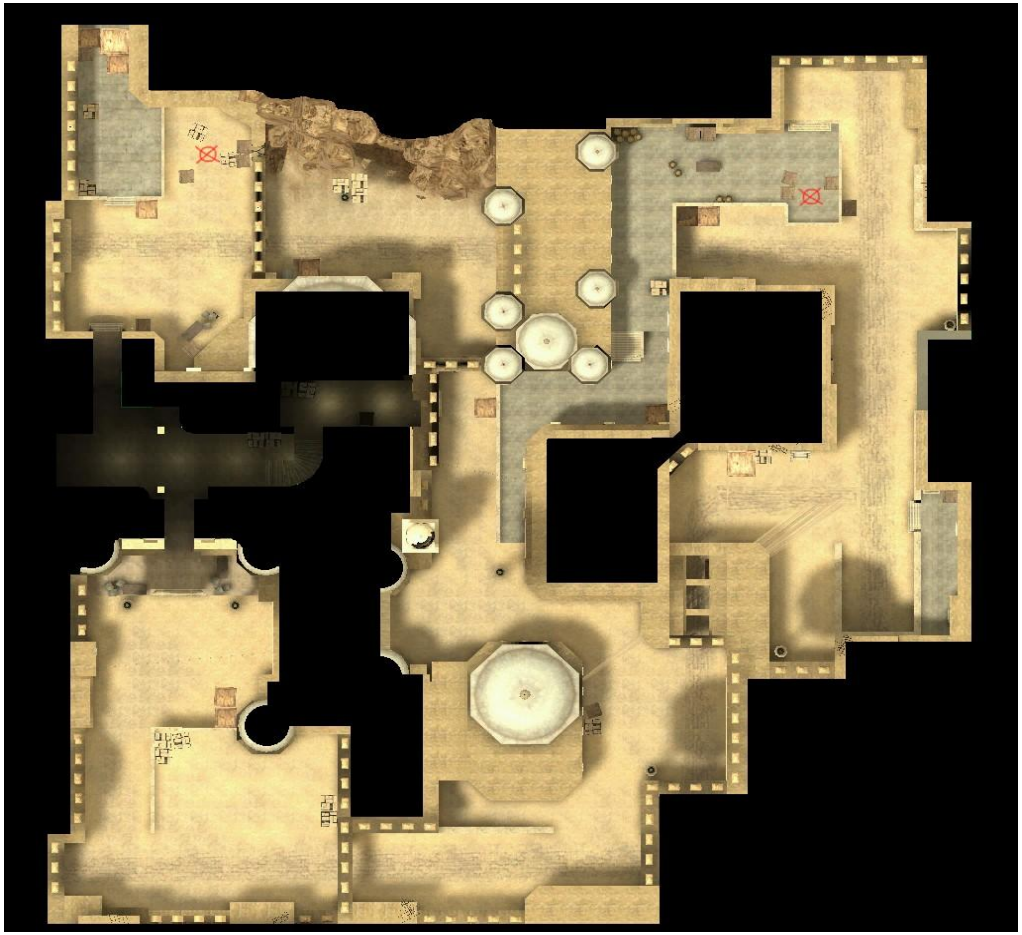
SimpleFPS domain

SimpleFPS domain

- Represent only very simple high-level features:
  - Game-world consists of interconnected **areas**, each of which has a number of **points of interest** (POIs).
  - A few types of **items** located at POIs (weapons, ammo, med-kits, keycards).
  - NPC can perform **basic actions** (move between areas or POIs, pick-up/use items, attack, take cover).

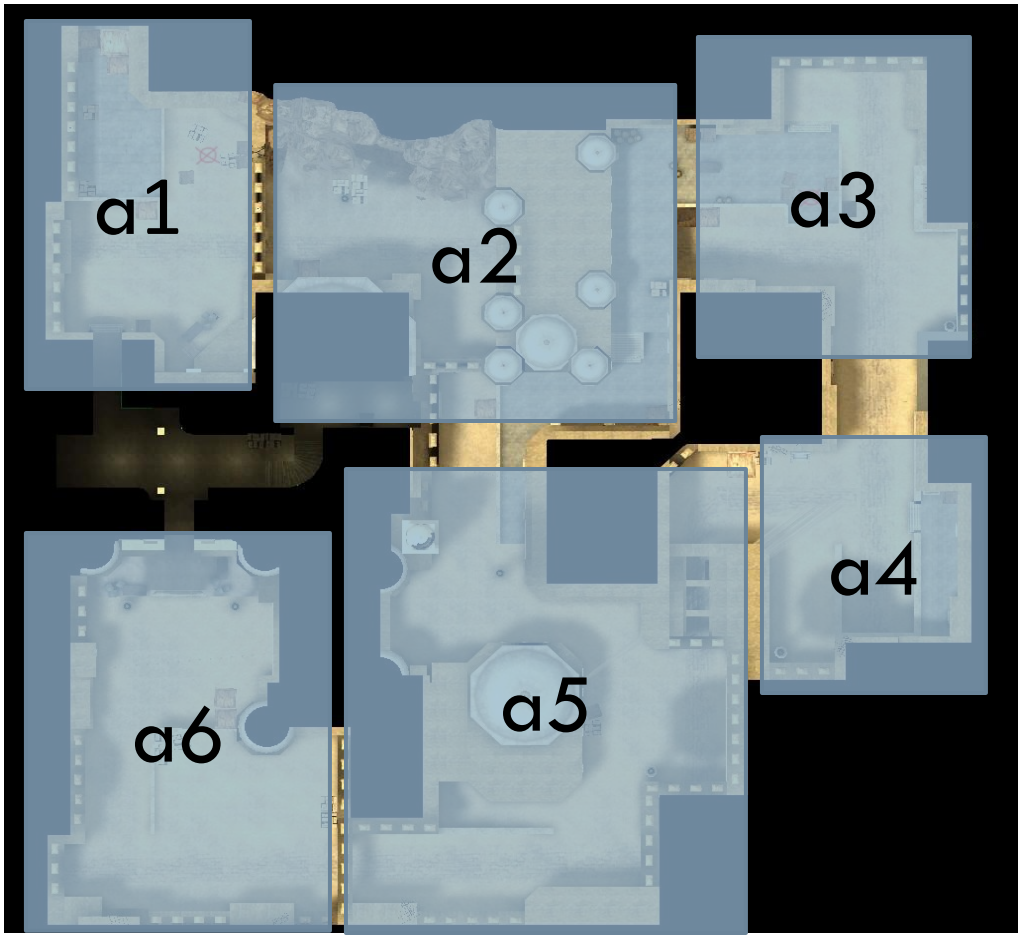
A SimpleFPS problem instance

- (:init …)



A SimpleFPS problem instance

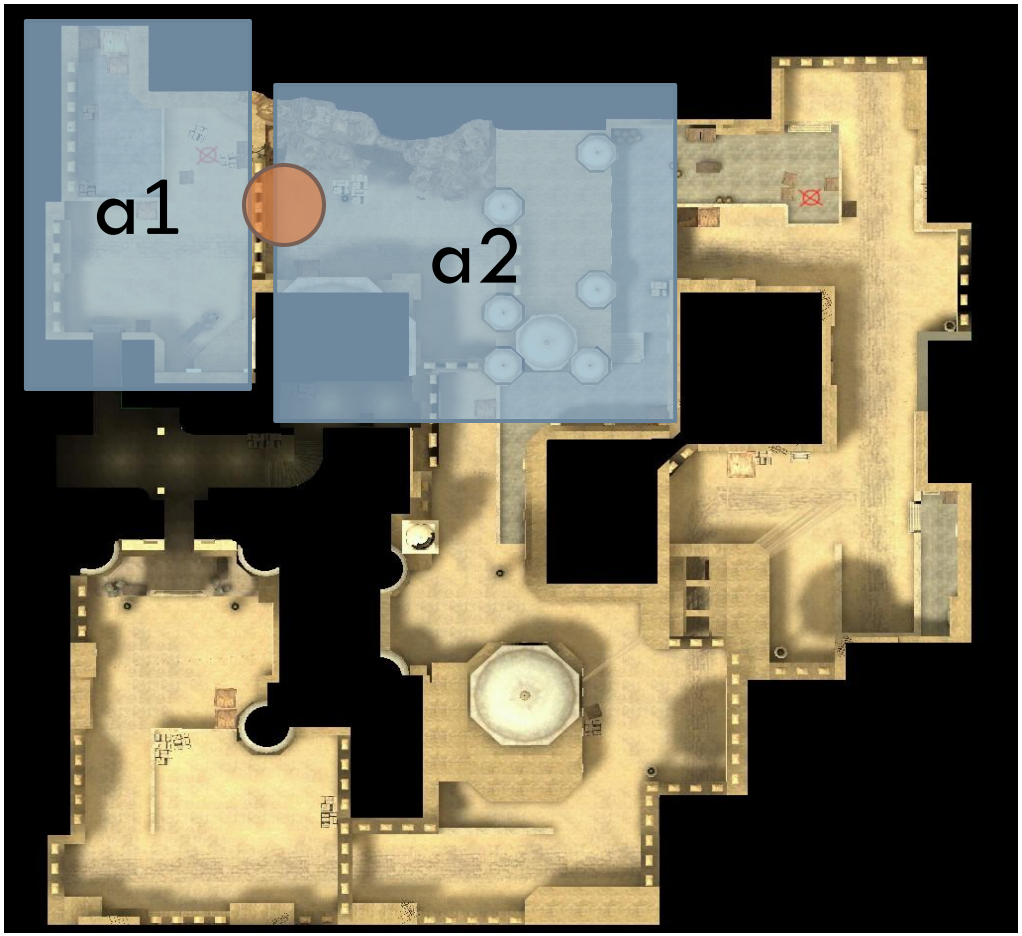
- (:init …)



- 6 areas, some of them connected through doors and corridors:
  - (area a1)
  - (area a2)
  - (area a3)
  - (area a4)
  - (area a5)
  - (area a6)

A SimpleFPS problem instance

- (:init …)

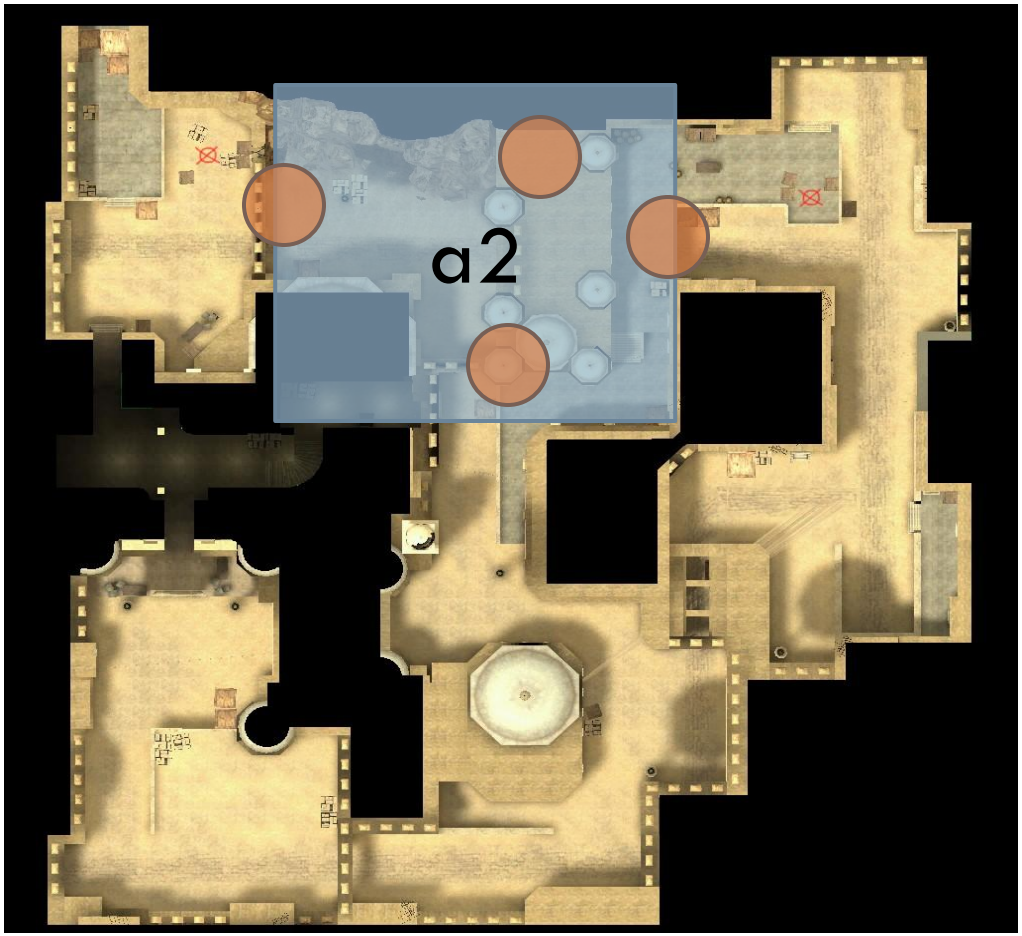


- 6 areas, some of them connected through doors and corridors:

- (poi door1 a1)
- (waypoint door1)
- (connected a1 a2 door1)
- (closed door1)
- (opens door1 keycard1)

A SimpleFPS problem instance

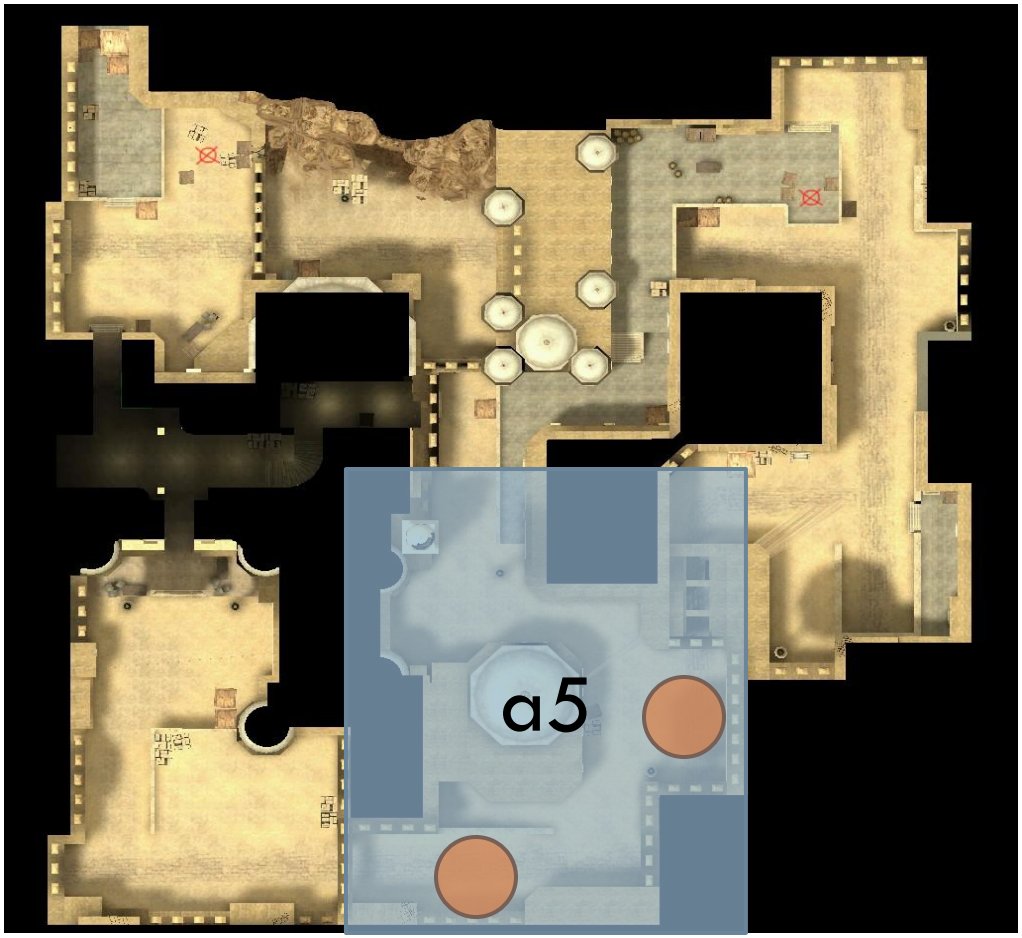
- (:init …)



- For each area a number of POIs are listed along with their properties:
- (poi door1 a2)
- (poi c1 a2)
- (poi c2 a2)
- (connected a2 a1 door1)
- (cover-point c1)
- (cover-point c2)

A SimpleFPS problem instance

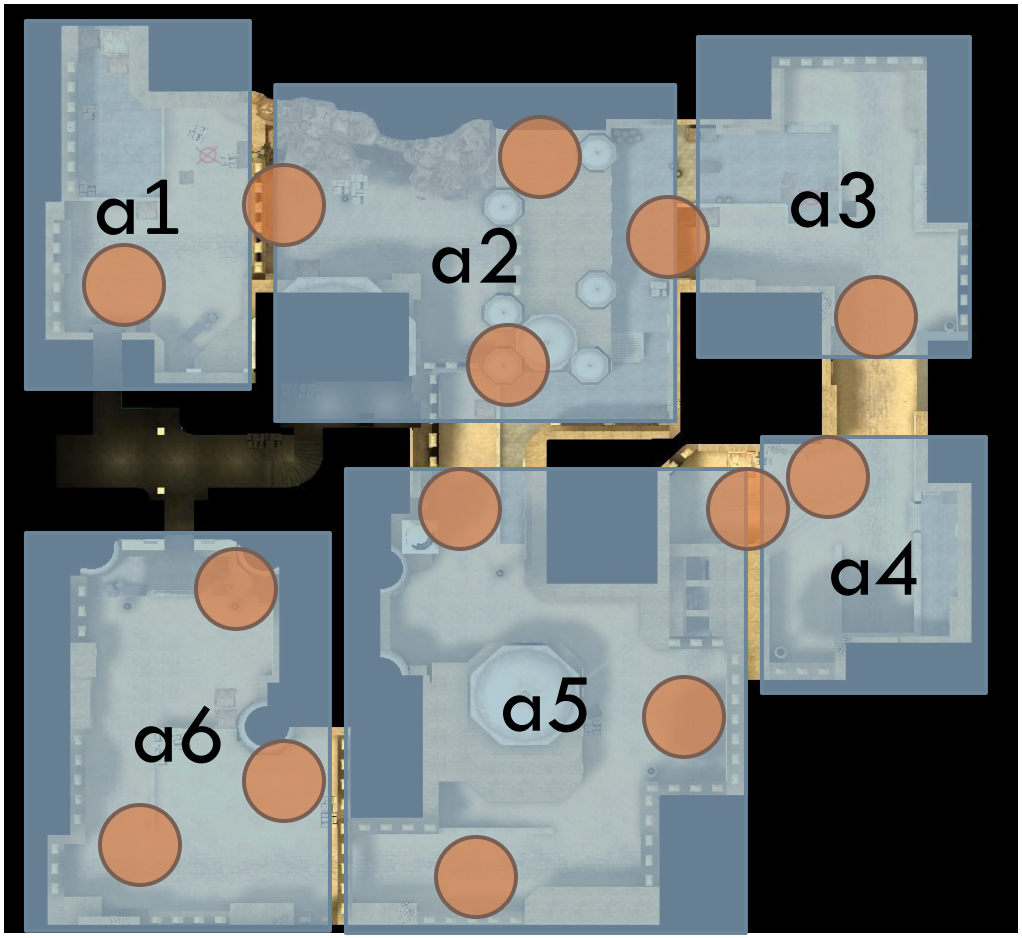
- (:init …)



- For each area a number of POIs are listed along with their properties:
- (poi g1 a5)
- (poi amm1 a5)
- (gun g1)
- (unloaded g1)
- (ammo amm1 g1)

A SimpleFPS problem instance

- (:init …)



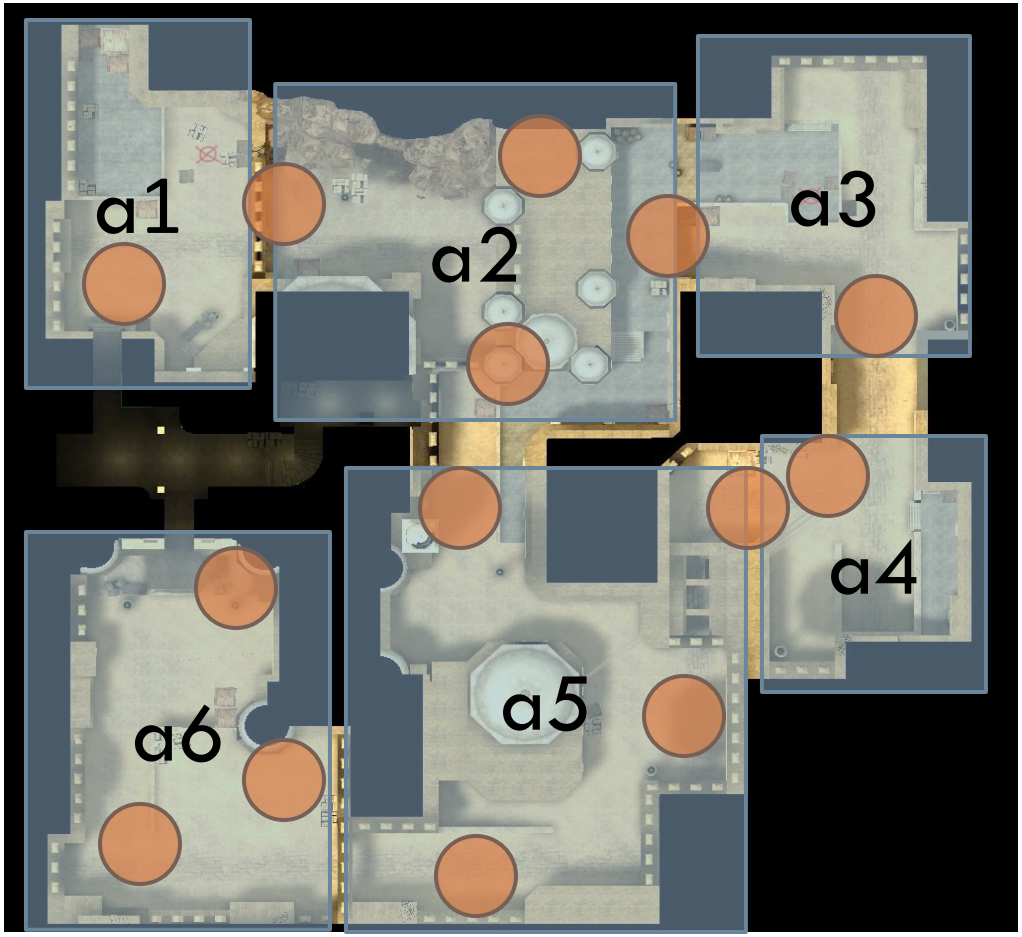
- For each area a number of POIs are listed along with their properties:
  - knife
  - med-kit
  - control-box
  - night-vision-gun
  - …

SimpleFPS domain: predicates

- NPC-related
  - (npc-at ?a)
  - (npc-close-to ?p)
  - (npc-covered)
  - (npc-uncovered)
  - (npc-holding ?o)
  - (npc-injured)
  - (npc-full-health)
  - (npc-aware)
  - (npc-unaware)
- Area-related
  - (area ?a)
  - (conn ?a1 ?a2 ?w)
  - (waypoint ?w)
  - (lighted ?area)
  - (dark ?area)
  - (poi ?p ?a)
  - (control-box ?p)
  - (cover-point ?p)
  - (item ?p)
- Item-related:
  - (med-kit ?m)
  - (knife ?k)
  - (gun ?g)
  - (loaded ?g)
  - (unloaded ?g)
  - (ammo ?i ?g)
  - (night-vision ?g)

SimpleFPS domain: actions

- (: action …)



- Available NPC actions:
  - move-to-area
  - move-to-poi
  - pick-up-item
  - use-item
  - take-cover
  - un-cover

SimpleFPS domain: move-to-point


```
(:action move-to-point
   :parameters (?area ?point)
   :precondition (and
                  (npc-at ?area)
                  (point-of-interest ?point ?area)
   )
   :effect (and
                  (npc-close-to ?point)
   )
)
```

SimpleFPS domain: reload

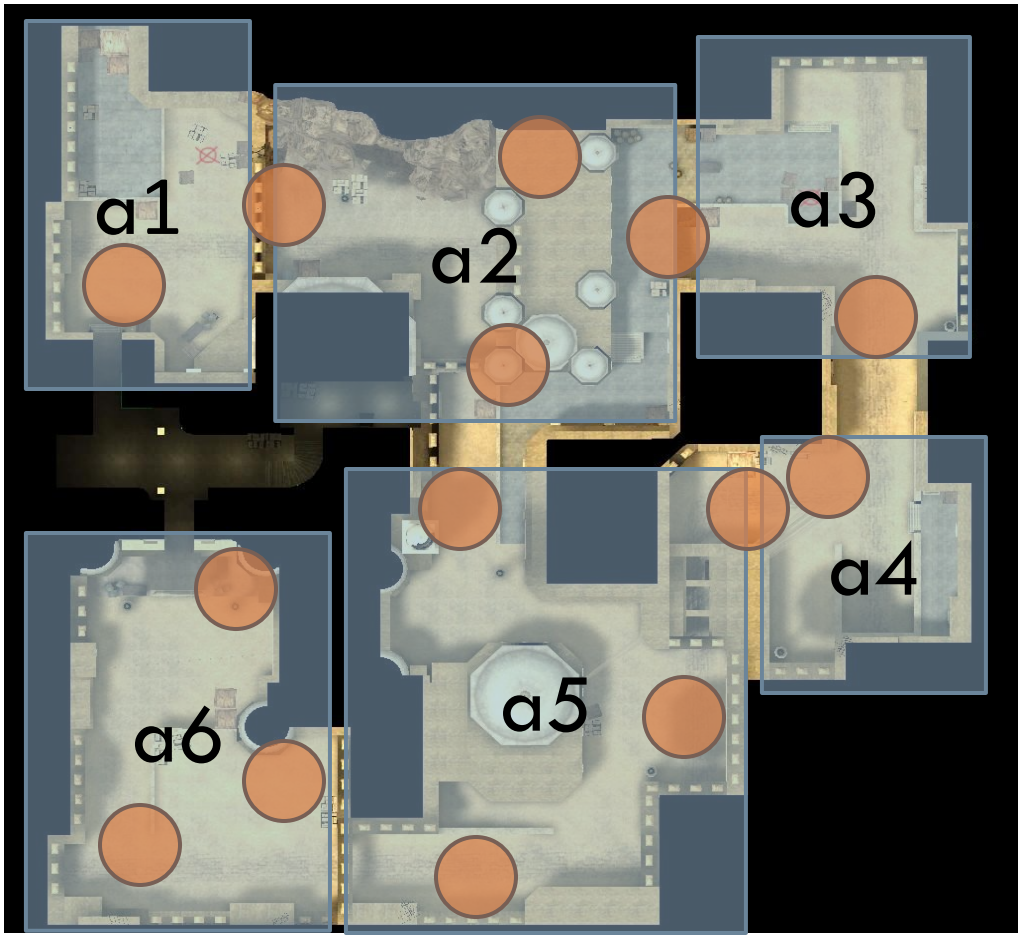

```
(:action reload
   :parameters (?gun ?item)
   :precondition (and
                  (npc-holding ?gun) (gun ?gun) (unloaded ?gun)
                  (npc-holding ?item) (ammo ?item ?gun)
   )
   :effect (and
                  (not (unloaded ?gun))
                  (loaded ?gun)
                  (not (npc-holding ?item))
   )
)
```

The SimpleFPS domain: actions

- Location-related:
  - moving-to-patrol
  - moving-to-take-position
  - move-away-from-point
  - move-to-point
  - move-to-point-from-point
  - make-accessible
  - place-in-inventory
  - turn-on-lights
  - turn-off-lights
- Attack-related:
  - make-contact
  - take-cover
  - uncover
  - use-med-kit
  - reload
  - attack-melee
  - attack-ranged
  - sneak-kill

A SimpleFPS problem instance

- (:goal …)



- NPC goals:
- **g1:** (player-wounded)
- **g2:** (npc-covered)
- **g3:** (npc-full-health)
- **g4:** (and g1 g2 g3)

A SimpleFPS problem instance

- blackbox -o problems\sfps-domain.txt -f problems\sfps-problem1.txt

1. (move-to-point area3 door3-2)
2. (move-to-point area2 control-box2)
3. (turn-on-lights area2 control-box2)
4. (make-contact area2 p)
5. (move-to-point-from-point area2 knife2 control-box2)
6. (place-in-inventory area2 knife2)
7. (move-to-point area2 p)
8. (attack-melee area2 knife2 p)
9. (move-to-point-from-point area2 door2-0 p)
10. (moving-to-take-position area2 area0 door2-0)
11. (move-to-point area0 coverpoint1)
12. (take-cover area0 coverpoint1)

SimpleFPS problem generator

SimpleFPS problem generator

- Takes as input:
  - -a number of areas,
  - -c the probability that two areas are connected
  - -n total number of points of interest
  - -g the goal condition as one of g1, g2, g3, g4
  - -l the number of instances to be generated
- Generates problem instances also using some rules:
  - Card-keys are added for locked doors
  - Ammo is added for guns that are unloaded
  - …

SimpleFPS datasets

- Used the tool to generate 3 datasets:
  - 5 areas
  - 7 areas
  - 10 areas
- For each dataset we generated 10 instances with:
  - 10 items
  - …
  - 100 items
- For each of the 4 goals:
  - g1, g2, g3, g4

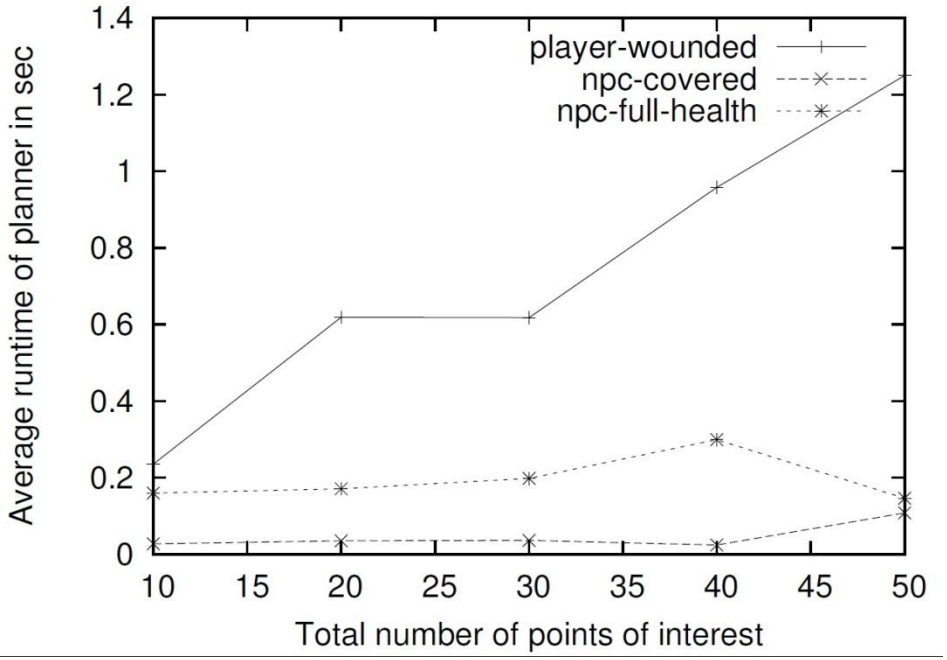
Preliminary results with SimpleFPS

- Used two award-winning planners in these datasets:
  - BlackBox [Kauts, Selman 1999]
  - FastForward [Hoffman 2001]
- Run the planners on an average laptop
  - 1.4 GHz
  - 2 GB RAM

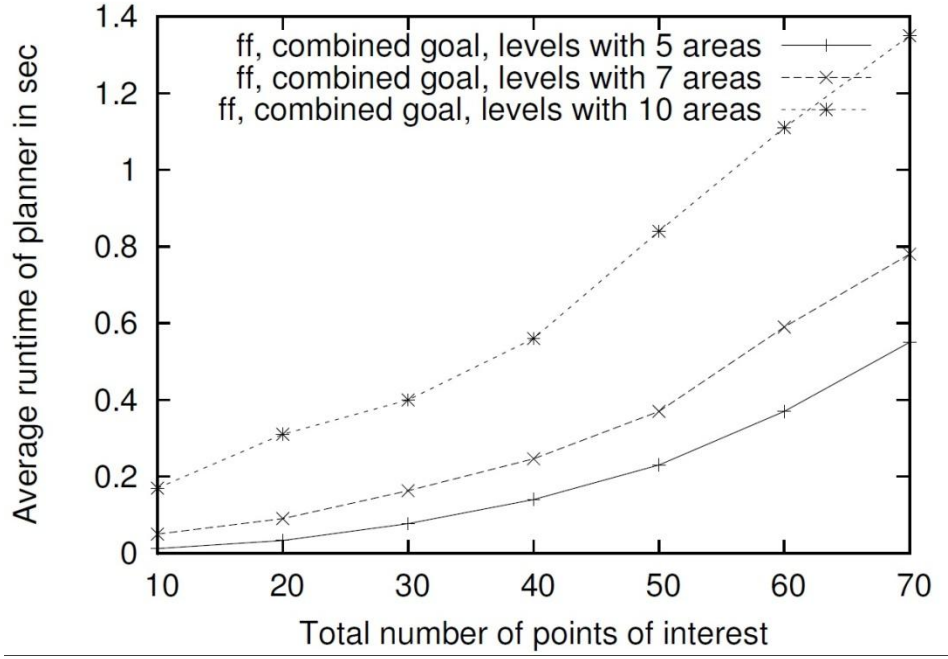
Preliminary results with SimpleFPS

- BlackBox, FastForward
  - Problems planners always return an answer within 1.5sec

BB: up to 5 areas/50 POIs



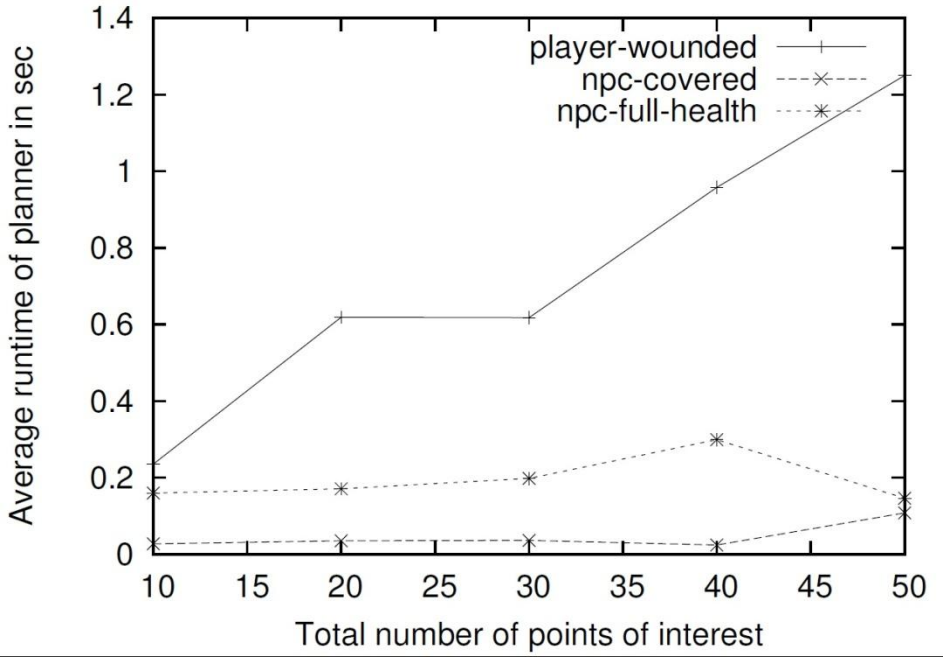
FF: up to 10 areas/70 POIs



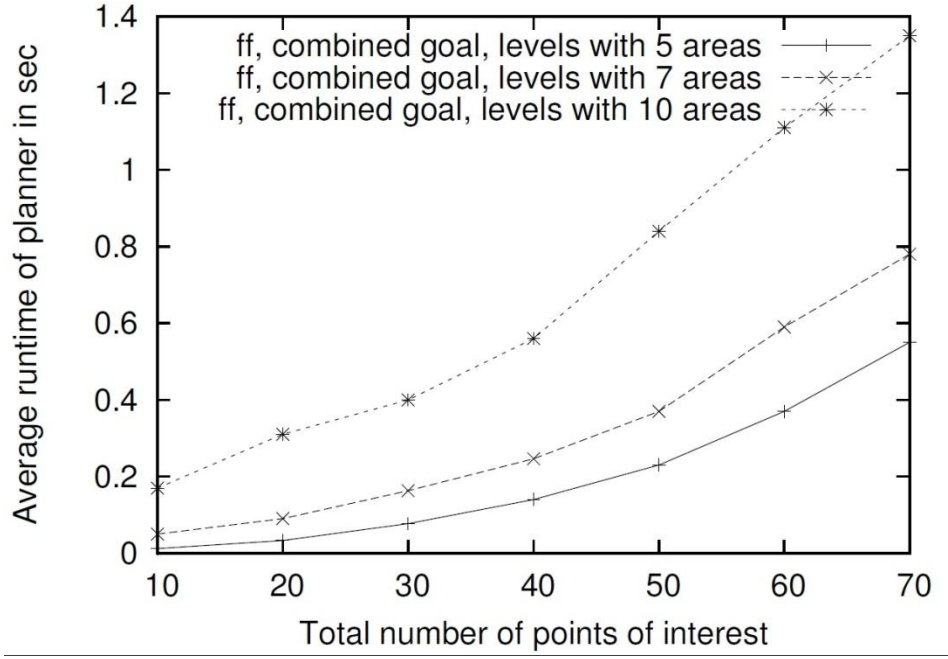
Preliminary results with SimpleFPS

Different planning techniques make a lot of difference

BB: up to 5 areas/50 POIs



FF: up to 10 areas/70 POIs

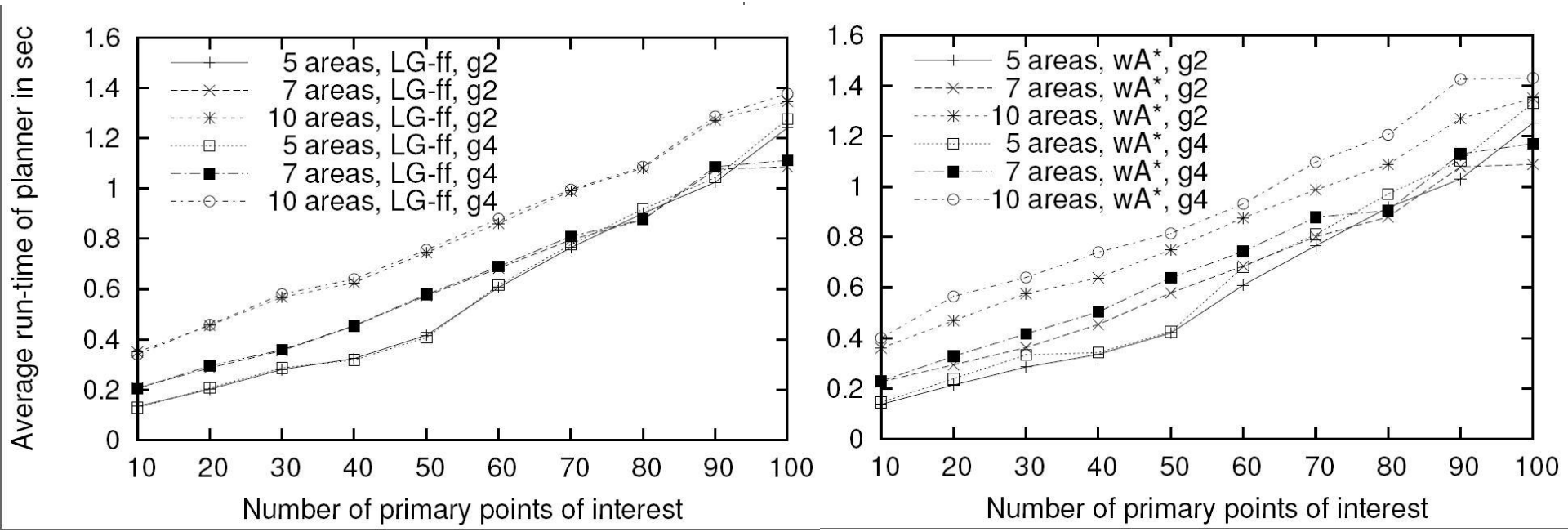


Preliminary results with SimpleFPS

- Investigate further the available heuristics for progression planning using FastDownward as a framework

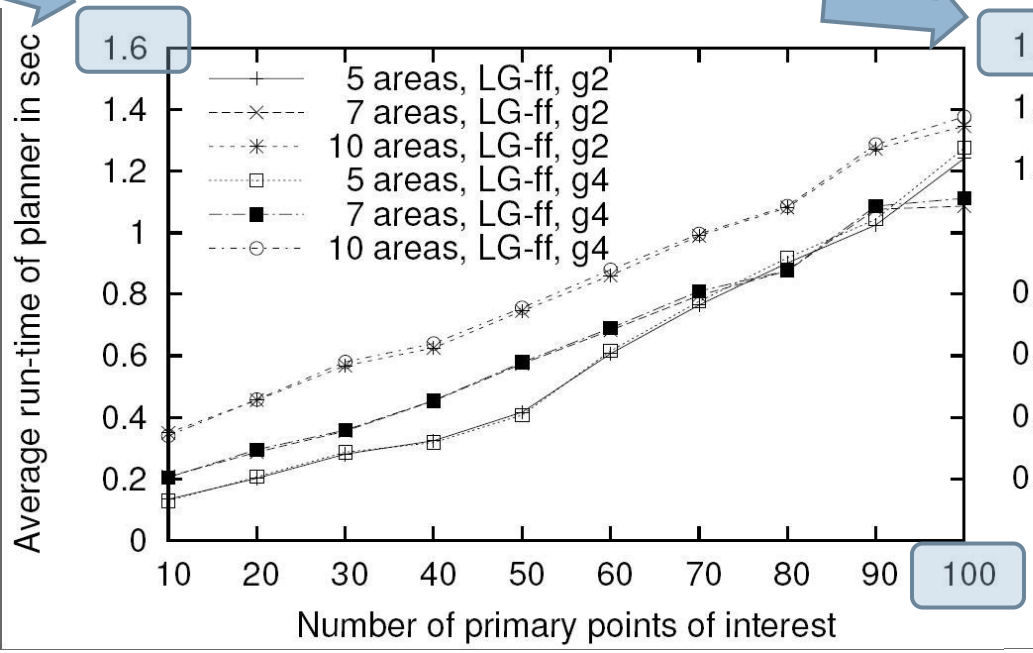
Thorough results with SimpleFPS

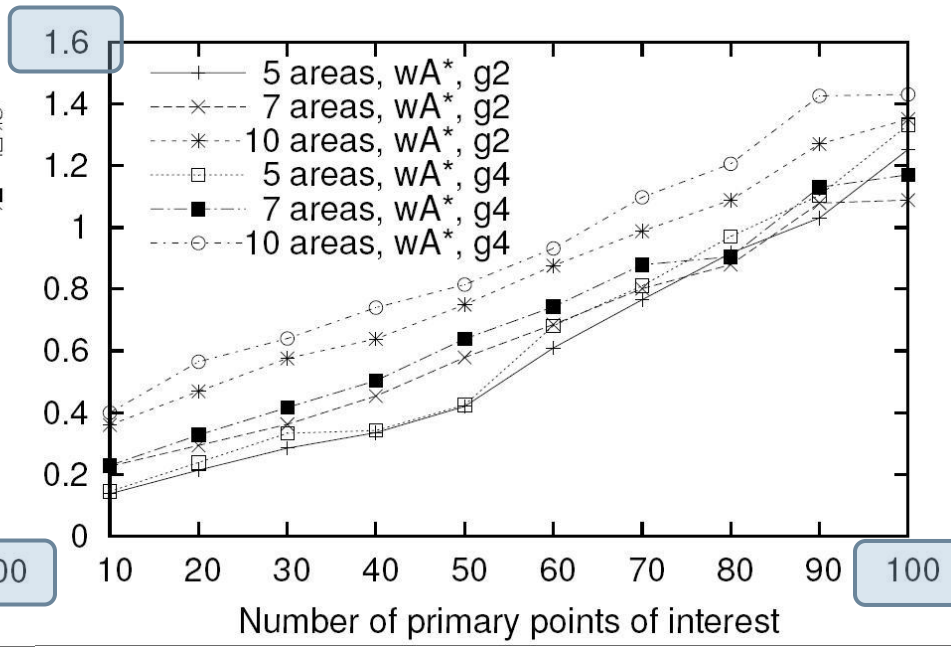
- Left: best-first search with FF heuristic
- Right: weighted A* search with a combination of FF and landmark cut heuristic
- Average (total) run-time of planner



Thorough results with SimpleFPS

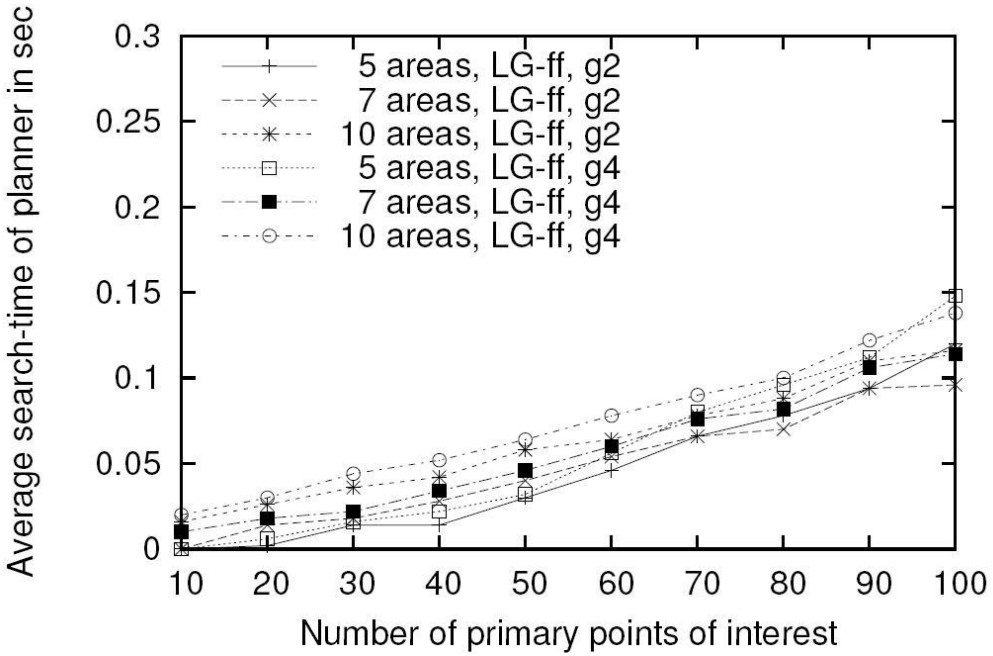
- Left: best-first search with FF heuristic
- Right: weighted A* search with a combination of FF and landmark cut heuristic
- Average (total) run-time of planner

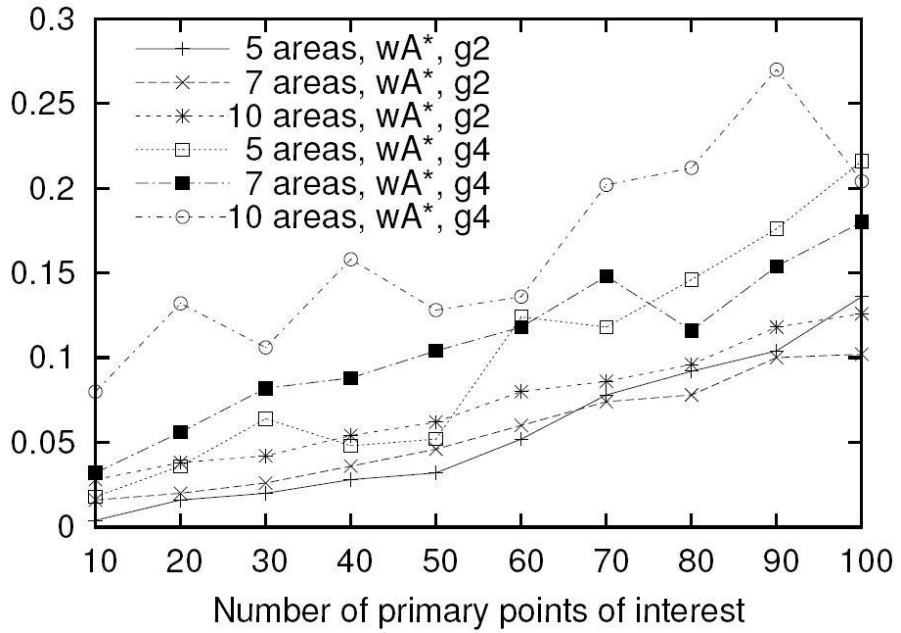




Thorough results with SimpleFPS

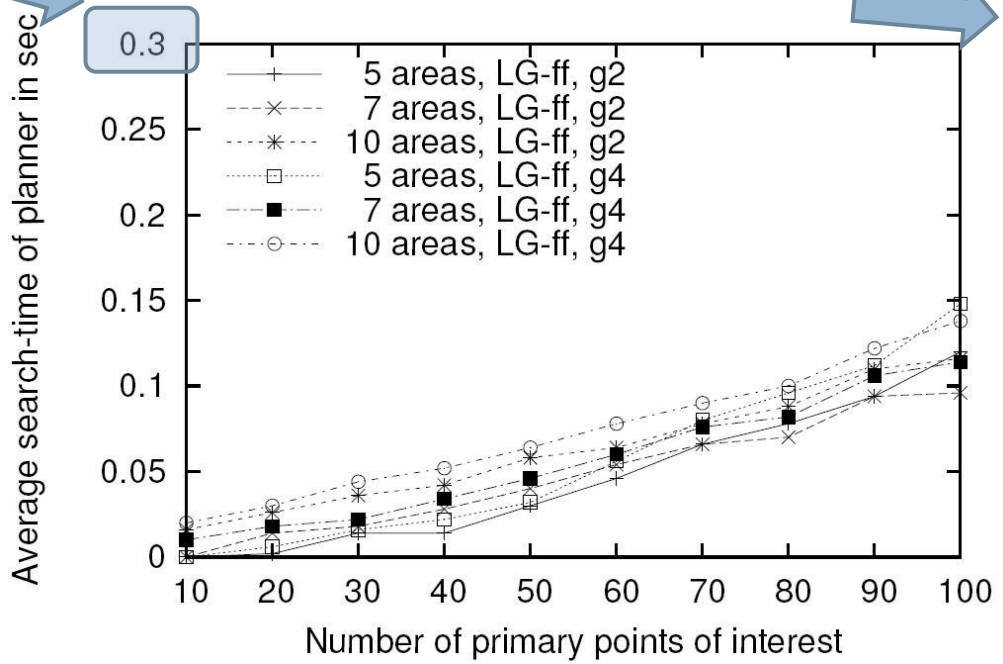
- Left: best-first search with FF heuristic
- Right: weighted A* search with a combination of FF and landmark cut heuristic
- Average run-time of planner doing search

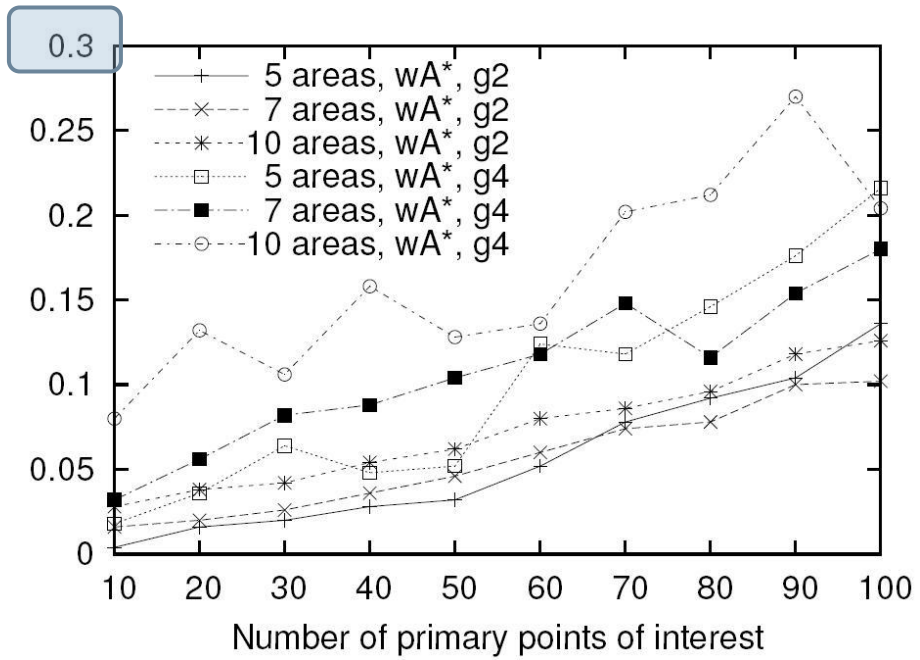




Thorough results with SimpleFPS

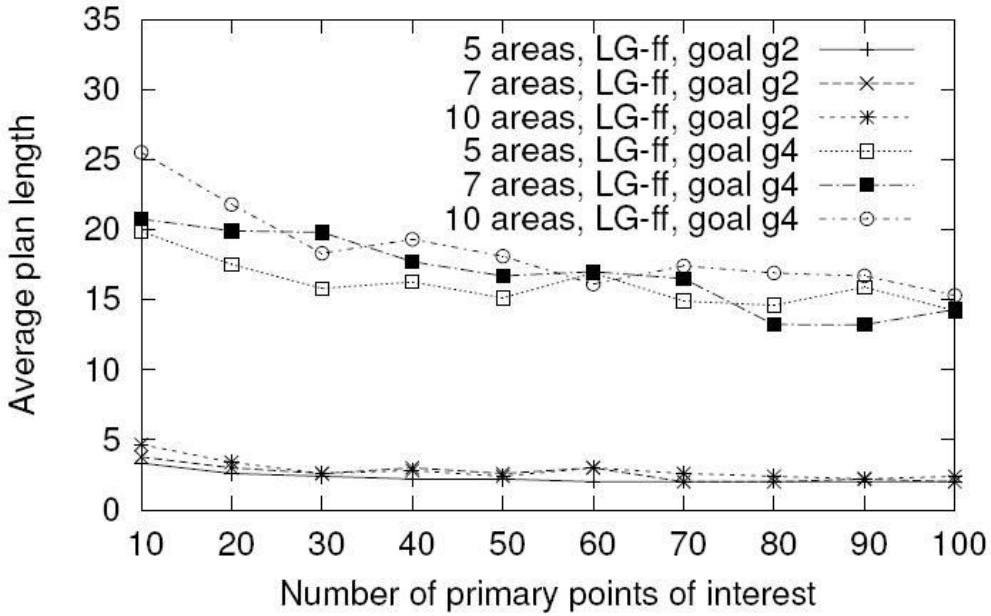
- Left: best-first search with FF heuristic
- Right: weighted A* search with a combination of FF and landmark cut heuristic
- Average run-time of planner doing search

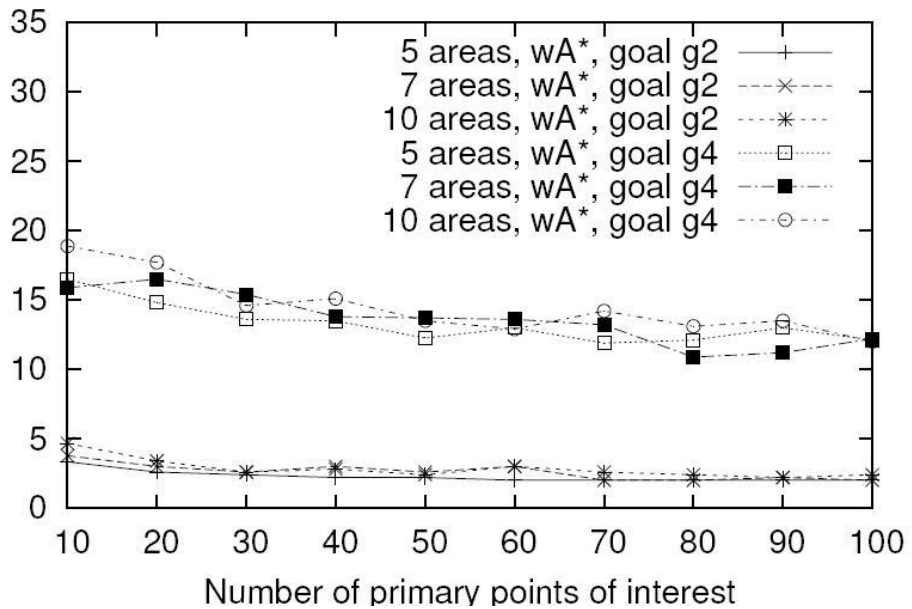




Thorough results with SimpleFPS

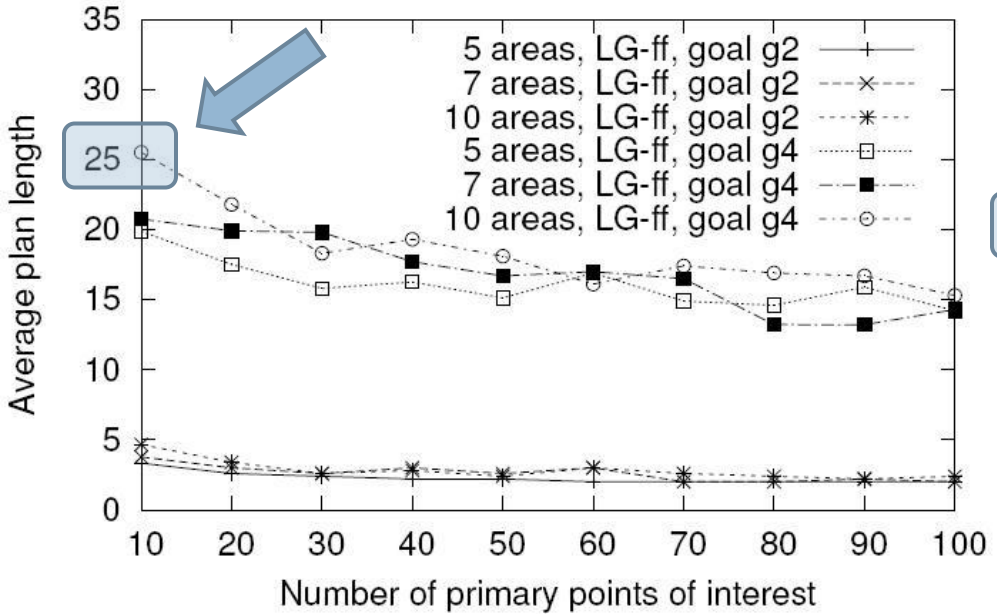
- Left: best-first search with FF heuristic
- Right: weighted A* search with a combination of FF and landmark cut heuristic
- Average plan length

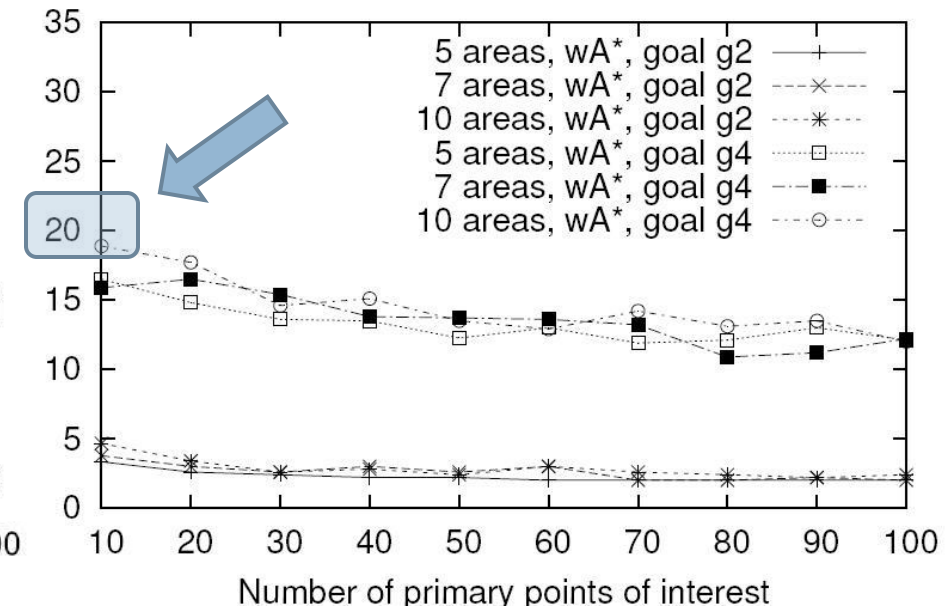




Thorough results with SimpleFPS

- Left: best-first search with FF heuristic
- Right: weighted A* search with a combination of FF and landmark cut heuristic
- Average plan length





Some conclusions

Some conclusions

- Relatively small-sized problems wrt FPS games:
  - 10 areas/100 POIs
- The planner takes a lot of resources:
  - A lot of **time**: up to 1.4 sec to respond using 100% of the CPU resources of a laptop
  - A lot of **memory**: up to 6.5MB for each problem

Some conclusions

- Relatively small-sized problems wrt FPS games:
  - 10 areas/100 POIs
- The planner takes a lot of resources:
  - A lot of **time**: up to 1.4 sec to respond using 100% of the CPU resources of a laptop
  - A lot of **memory**: up to 6.5MB for each problem
- **Different assumptions in academia and FPS games**

Some conclusions

- There is a lot of room for improvement
- Take advantage of pre-processing
  - 6.5 MBs is too much for 1 character but how about 100?
  - Maintain the pre-processed state and update instead of re-computing it each time
- Guide the search method
  - The Golog family of languages

Bibliography

- References

  - The SimpleFPS Planning Domain: A PDDL Benchmark for Proactive NPCs. Stavros Vassos, Michail Papakonstantinou. In Proceedings of the Non-Player Character AI workshop (NPCAI-2011), 2011
  - PDDL - The Planning Domain Definition Language. Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, David Wilkins. Technical report, Yale Center for Computational Vision and Control, TR-98-003, 1998.
  - Unifying SAT-Based and Graph-Based Planning. Henry Kautz, Bart Selman. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 1999
  - The FF planning system: Fast plan generation through heuristic search. Jörg Hoffmann, Bernhard Nebel. Artificial Intelligence Research, Vol. 14, 2001
  - The Fast Downward Planning System. Malte Helmert. Artificial Intelligence Research (JAIR), Vol. 26, 2006