IPC 2023 - Learning Track

Jendrik Seipp & Javier Segovia-Aguas

Setup

- like 2008, 2011, 2014, but all computations done by organizers
- only bugfixes allowed after submission deadline
- goal: reproducibility
- submit two Apptainer files

Learning

./learn dk DOMAIN TASK1 TASK2 TASK3 ...

produces files `dk.1`, `dk.2`, etc.

Planning

./plan dk.5 DOMAIN TASK plan

finds plans `plan.1`, `plan.2`, etc.

Planning Tasks

- STRIPS, unit costs, types, negative preconditions
- Training: $\sim$99 “easy” instances, out of which $\sim$10 handwritten base cases
- Testing: 30 easy, 30 medium and 30 hard instances
- Experiments on instances:
  - base cases can be fully expanded,
  - some easy cases are solvable with blind search,
  - many easy cases can be solved optimally with LM-cut,
  - most easy cases and many medium are solvable with LAMA
- A plan is generated for each instance with a domain-dependent strategy, and validated with both the Unified Planning and the Universal Planning Validator frameworks.
- All tasks and code available online.

Environment

Single-Core

- 1 CPU core (from an Intel Xeon Gold 6130 CPU), no GPU
- Limits training per domain: 24 hours, 32 GiB
- Limits evaluation per task: 30 minutes, 8 GiB

Multi-Core canceled

Metrics

- Quality score: C*/C
  Bounds C* obtained with domain-specific solvers, IPC planners, LAMA (8h, 32 GiB)
- Agile score: 1 - log(T)/log(1800)
- Same ranking $\rightarrow$ awards only for quality score

Baselines

- **Fast Downward SMAC 2014**
  Jendrik Seipp, Silvan Sievers, Frank Hutter
  Single Fast Downward configuration, optimized for minimal runtime with SMAC.
- **Progressive Generalized Planner**
  Javier Segovia-Aguas, Sergio Jiménez, Laura Sebastiá, Anders Jonsson
  Fixed configuration of PGP for the given training tasks.

Participants 1/2

- **ASNets 2023**
  Mingyu Hao, Ryan Wang, Sam Toyer, Felipe Trevizan, Sylvie Thiébaux, Lexing Xie
  Action Schema Networks implemented in Tensorflow 2.
- **GOFAI**
  Alvaro Torralba, Daniel Gnad
  Good Old-Fashioned AI that learns how to partially ground tasks from a given domain.
- **HUZAR**
  Piotr Rafal Gzubicki, Bartosz Piotr Lachowicz, Alvaro Torralba
  Learn to distinguish between good and bad transitions by feeding problem description graphs into a GNN.

Participants 2/2

- **Muninn**
  Simon Ståhlberg, Blai Bonet, Hector Geffner
  Learn relational message-passing neural networks for STRIPS.
- **NPGP**
  Chao Lei, Nir Lipovetzky, Krista A. Ehinger
  Novelty-based generalized planner that prunes a newly generated planning program if its most frequent action repetition is greater than a given bound.
- **Vanir**
  Dominik Drexler
  Learn width-based hierarchical policies for polynomial domains.

Domains and Results

- Muninn only system to not crash immediately in first tests
- Future: pass time and memory limits to Apptainer scripts

- PGP fails to learn DK in the evaluation domains

$\rightarrow$ NPGP fails as well, omit from results below

- Muninn is optimized to run on GPUs
- results below use only CPUs

Blocksworld

- **Description**: convert initial configuration of towers of n blocks into a goal configuration

- **Hardness**: 2-approximable (Gupta and Nau, AAAI 1991)

- **Strategy**:
  1. unstack all blocks (and the ones above) that are not at their goal location,
  2. pick and stack blocks as they appear in the goal

- **Parameter ranges**:
  - Easy: $n \in [5, 30]$
  - Medium: $n \in [35, 150]$
  - Hard: $n \in [160, 500]$

Childsnack

- **Description**: make s sandwiches in a kitchen either with gluten or gluten-free ingredients. Deliver sandwiches on t trays to the tables with c children (a are allergic to gluten).
- **Hardness Hypothesis**: PO
- **Strategy**:
  1. make c sandwiches, making as many gluten-free as possible, and the rest with gluten,
  2. put all c sandwiches on one tray, and move that tray from the kitchen to the first table
  3. for each child at the table, serve a sandwich with gluten if possible, otherwise serve a gluten-free sandwich
  4. move the tray to the next table with children and repeat the previous step, until all children are served
- **Parameter ranges**:
  - Easy: $c \in [4, 10]$, $a \in [0, 6]$, $t \in [1, 3]$, $s \in [4, 15]$
  - Medium: $c \in [15, 40]$, $a \in [15, 25]$, $t \in [2, 5]$, $s \in [15, 60]$
  - Hard: $c \in [50, 300]$, $a \in [50, 150]$, $t \in [4, 10]$, $s \in [50, 450]$

Ferry

- **Description**: c cars are randomly distributed into l locations, and a ferry with capacity for 1 car must transport them to their destinations.

- **Hardness Hypothesis**: 2-approximable

- **Strategy**:
  1. for each car in the goal, sail the ferry to its origin and board it
  2. sail the ferry to the car goal location and debark it, then repeat from step 1. until all goals are satisfied

- **Parameter ranges**:
  - Easy: $c \in [1, 20]$, $l \in [5, 15]$
  - Medium: $c \in [10, 100]$, $l \in [20, 50]$
  - Hard: $c \in [200, 1000]$, $l \in [100, 500]$

Floortile

- **Description**: a grid of $x \times y$ tiles has r robot painters, each in a different column, that can paint either the tile above or below with black/white color. Robots may move in 4 directions via unpainted tiles. All tiles except the bottom row must be painted in checkerboard style.
- **Hardness Hypothesis**: 2-approximable
- **Strategy**:
  1. move all robots (from left- to right-most) adjacent to each in the upper-left corner,
  2. if necessary, change to white color if robot coordinates (i, j) add up to an odd number, otherwise to black,
  3. move a robot down, paint its tile above, and swap colors, and repeat this for each robot until reaching the bottom row
  4. move the rightmost robot once to the right, then to the topmost tile, and repeat from step 2. only for this robot and until no more columns are left
- **Parameter ranges**:
  - Easy: $x, y \in [3, 8]$, $r \in [1, 3]$
  - Medium: $x, y \in [10, 22]$, $r \in [4, 15]$
  - Hard: $x, y \in [25, 37]$, $r \in [15, 35]$

Miconic

- **Description**: there are p passengers randomly distributed on f floors; an elevator (with ∞ capacity) that can board passengers "only" from their origin floor and let them depart "only" at their destination; and the elevator can move between any two floors.
- **Hardness**: 2-approximable (Helmert *et al.*, ECAI 2006)
- **Strategy**:
  1. move the elevator to the first floor with a passenger,
  2. board all passengers on that floor, and depart all the ones that are at their destination,
  3. move upwards to the next floor with a passenger to board or depart, and go back to step 2.; repeat until no more passengers to board or depart above,
  4. repeat the previous step but move the elevator down
- **Parameter ranges**:
  - Easy: $p \in [1, 10]$, $f \in [4, 20]$
  - Medium: $p \in [20, 80]$, $f \in [30, 60]$
  - Hard: $p \in [50, 500]$, $f \in [80, 200]$

Rovers

- **Description**: r rovers are equipped for analyzing soil/rock and/or taking images from o objectives with up to c cameras (requiring calibration), which must be communicated to a lander. Each soil and rock to analyze is in one of the w waypoints, and the objectives and the lander are visible from a subset of waypoints. Rovers can only navigate through a subset of waypoint edges that must be visible.
- **Hardness**: poly-APX (Helmert *et al.*, ECAI 2006), bounded plans found in polytime
- **Strategy**:
  1. for each rock/soil data in the goal, get a rover equipped for rock/soil analysis and can move to that waypoint, sample and drop it
  2. for each image in the goal, get a rover that can reach a waypoint to take the image and that has a camera that supports the corresponding mode. Move the rover to the corresponding waypoint to calibrate the camera, then to the waypoint to take the image,
  3. communicate all data after moving each rover to a waypoint where a lander is visible
- **Parameter ranges**:
  - Easy: $r \in [1, 4]$, $w \in [4, 10]$, $c \in [1, 4]$, $o \in [1, 10]$
  - Medium: $r \in [5, 10]$, $w \in [15, 90]$, $c \in [5, 50]$, $o \in [15, 80]$
  - Hard: $r \in [15, 30]$, $w \in [100, 200]$, $c \in [60, 100]$, $o \in [100, 200]$

Satellite

- **Description**: i switched-off instruments are on board s satellites and can take images in up to m modes. Satellites point and turn to any of the d directions. Only one instrument can be active at a time in a satellite, and they need to calibrate in a specific direction when they are switched on before taking images.
- **Hardness**: 6-approximable (Helmert *et al.*, ECAI 2006)
- **Strategy**:
  1. for a goal image, switch on the instrument in a satellite that supports the goal mode
  2. turn the satellite to calibration target if necessary, and calibrate the instrument
  3. turn to goal direction and take the image, switch it off and repeat from step 1. until there are no more images to take
  4. for each goal pointing direction, turn the satellite to that direction if necessary
- **Parameter ranges**:
  - Easy: $s \in [3, 10]$, $i \in [3, 20]$, $m \in [1, 3]$, $d \in [4, 10]$
  - Medium: $s \in [15, 40]$, $i \in [15, 80]$, $m \in [3, 5]$, $d \in [15, 30]$
  - Hard: $s \in [50, 100]$, $i \in [50, 200]$, $m \in [5, 10]$, $d \in [40, 100]$

Sokoban

- **Description**: an agent in a $g \times g$ grid, with some locations blocked by walls, must push b boxes (in any of the 4 cardinal directions) to their goal.

- **Hardness**: PSPACE-complete (Culberson, 1997)

- **Strategy**: no polynomial approximation, so solvable instances are generated first by moving the agent to each box iteratively, and pushing them up to a maximum number of moves, every unvisited location then becomes a candidate to place a wall.

- **Parameter ranges**:
  - Easy: $g \in [8, 13]$, $b \in [1, 4]$
  - Medium: $g \in [20, 50]$, $b \in [5, 35]$
  - Hard: $g \in [60, 100]$, $b \in [40, 80]$

Spanner

- **Description**: an agent can only move forward from a shed to a gate, by crossing l locations of a corridor, and collect up to s spanners to tighten all n loose nuts at the gate. Spanners break when they are used.

- **Hardness Hypothesis**: PO

- **Strategy**:
  1. move to the next location,
  2. collect all spanners in that location up to a total of n spanners, and repeat from step 1. until reaching the gate,
  3. use each collected spanner to tighten a loose nut, and repeat until all nuts are tightened

- **Parameter ranges**:
  - Easy: $s \in [1, 10]$, $n \in [1, 5]$, $l \in [4, 10]$
  - Medium: $s \in [30, 90]$, $n \in [15, 50]$, $l \in [15, 45]$
  - Hard: $s \in [100, 500]$, $n \in [50, 250]$, $l \in [50, 100]$

Transport

- **Description**: p packages are randomly distributed to l strongly connected locations, and need to be delivered to other destination locations by using up to v vehicles with m maximum capacity.

- **Hardness Hypothesis**: poly-APX

- **Strategy**:
    1. using always the same vehicle, for each package in the goal, drive from vehicle location to package starting location (path search), and pick it up,
    2. drive from package origin to its destination (path search), and drop it, and repeat from step 1. until no more packages to transport

- **Parameter ranges**:
    - Easy: $v \in [3, 6]$, $p \in [1, 15]$, $l \in [5, 15]$, $m = 2$
    - Medium: $v \in [10, 20]$, $p \in [5, 45]$, $l \in [20, 40]$, $m = 4$
    - Hard: $v \in [30, 50]$, $p \in [20, 200]$, $l \in [50, 100]$, $m = 10$

Quality Scores

	Baselines			Competitors				
	LAMA	FDSS	SMAC	ASNets	GOFAI	HUZAR	Muninn	Vanir
Blocksworld	47.9	49.4	31.5	4.6	**46.4**	39.3	40.6	–
Childsnack	26.2	35.4	20.2	0.0	**26.5**	22.0	11.0	–
Ferry	64.0	61.5	64.4	–	58.5	58.7	42.1	**76.3**
Floortile	12.0	22.7	24.7	–	**34.4**	21.3	0.0	–
Miconic	84.4	89.6	52.3	7.2	**81.4**	72.4	30.0	75.2
Rovers	66.8	64.0	58.1	6.5	54.4	60.0	14.2	**66.1**
Satellite	87.3	88.7	71.0	–	74.0	79.9	16.0	**87.3**
Sokoban	37.7	39.0	30.8	0.0	**38.4**	28.1	24.3	37.7
Spanner	30.0	60.7	30.0	8.9	30.0	30.0	**32.0**	–
Transport	61.4	63.0	62.7	2.0	**64.5**	55.4	16.2	–
Sum	517.6	574.1	445.7	29.1	**508.5**	467.0	226.3	342.6

Winners

Runner-Up

HUZAR by Piotr Rafal Gzubicki, Bartosz Piotr Lachowicz and Álvaro Torralba

Winner

GOFAI by Álvaro Torralba and Daniel Gnad