Accepted in ICAPS-'17 !

Improving Greedy Best-First Search by Removing Unintended Search Bias

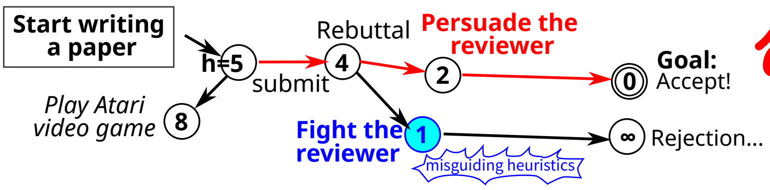
University of Tokyo
Masataro Asai, Alex Fukunaga

1. Backgrounds

Greedy Best First Search

- Satisficing search algorithm for finding a goal as quickly as possible (w/o optimality guarantee)
- Best-first search using goal estimate (h-value): expand the node w/ the smallest h-value in the OPEN list
- Search is sorely guided by the heuristic estimate h

→ May be misguided by incorrect heuristics



Diversified Greedy Best First Search

- Strategy to avoid the pathological behavior
- Occasionally ignore the best-h-first ordering

Plateau (also called Uninformative Heuristic Region)

- A set of nodes with the same h-value
- Indistinguishable in terms of priority
- Large plateau slows down the search

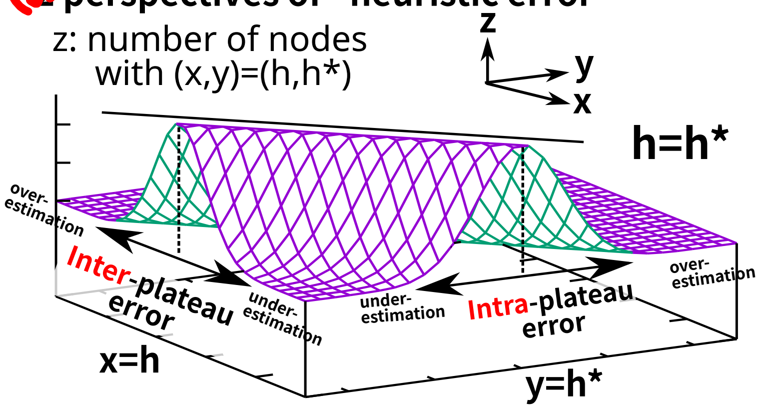
address

2. Intra/Inter Plateau Diversification

two

2 perspectives of "heuristic error"

z: number of nodes with (x,y)=(h,h*)

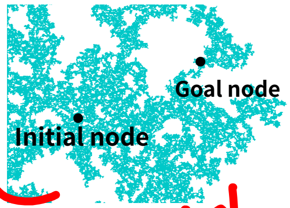


Nodes w/ same h*-value have different h-value →Naive GBFS always selects minimum h →low-h* nodes w/ high-h may not be expanded	Nodes w/ same h-value have different h*-value →Naive tiebreaking may keep expanding high-h* nodes
Inter-plateau Diversification: Randomized h-selection	**Intra-plateau Diversification: Randomized tiebreaking (=node selection in a certain h-plateau)**

- h-selection strategy and tiebreaking strategy after h-selection do not interfere each other
- Therefore, their effects are orthogonal

3. Invasion Percolation for Search Diversification

Invasion Percolation (IP) (Wilkinsonand Willemsen '83)



- Physical model for the distribution of fluid slowly invading porous media (e.g. water replacing the oil in a porous rock for retrieving oil)
- Theoretical characteristics are well studied in physics; Fractal dimension, Isotropic (rotational invariance)
- Node marking algorithm for IP is shown to equivalent to Prim's method for Minimum Spanning Tree (Barbasi, '96)

dense search

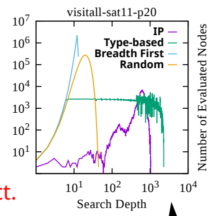
Using IP for diversified blind search

purpose

- Sort the search nodes according to r_{BIP} function and run best-first search

$$r_{BIP} = \min_{\forall e \in \text{parents}} \textit{memo}(e, \text{random}())$$

- r_{BIP}: minimum of the random values memoised on each search edge
- Embankment effect: high-r_{BIP} value nodes surround a certain region and prevent / delay the exploration inside it



- Basic random tiebreaking does not have this effect.

random selection
== pick the first node of the results of random sort
== allocate new random values to the nodes without memo
→ new value may allow for exploration inside embankment

Distribution of the nodes per depth in blind search

- **Breadth First ~ Random, while IP shows a significantly lower branching factor because it skips expanding large number of nodes & tries to explore as far as possible from the initial state**

Using IP for diversified GBFS

Can be used as both Intra- and Inter-plateau tiebreaking

- Intra-plateau: use r_{BIP} for breaking the tie in GBFS
- Inter-plateau: Alternate nomal GBFS (best-h-first) expansion with expansion from an additional queue sorted by r_{BIP}

5. Evaluation

Exp1: The perfromance of the same algorithm as Intra/Inter-plateau div. (hd (randomdepth tiebreaking, Asai '16) vs hD (=Type-based, Xie'14) --- both are Type-based):
→ orthogonal improvements were observed

diversification wrto depth vs. wrto width

Exp2: Combining Inter+Intra (hdD=hd+hD):
→ Better performance than either of single variants (hd,hD)
Exp3: IP diversification (hb,hB,hbB=hb+hB):
→ Better performance in different domains than hd/hD/hdD

5 min, 4GB, 10 runs, all based on FD. Number of solved instances

		h^{CG} h	h^{CG} hd intra	h^{CG} hD inter	h^{CG} hdD both	h^{FF} h	h^{FF} hd intra	h^{FF} hD inter	h^{FF} hdD both
	total	187	194.2	206.1	215.8	192	208	207.4	223.9
IPC11 w/o duplicates	elevators	9	8	8.7	9.7	19	14	15.9	13.7
	nomystery	7	6	15.4	15.1	9	7	16.6	17
	parcprinter	20	20	19.4	18.7	20	20	20	20
	pegsol	20	20	20	20	20	20	20	20
	scanalyzer	20	20	19.9	20	15	15.1	18	18.6
	sokoban	16	16	16.9	17	19	19	17.4	17.4
	tidybot	16	18	18.7	18.6	16	16	16	16.7
	woodwork	2	2	2.7	7.7	2	2	4	7.2
IPC14	barman	0	0	0	0	0	0	1.5	1
	cavediving	7	7	7	7	7	7	7	7.2
	childsnack	1	6	0.1	1.5	0	4	0	0.3
	citycar	0	0	7.8	4.7	0	0	7.2	7.1
	f oortile	0	0	2	2	2	2	2	2.1
	ged	0	0	9.6	9.7	19	19	14	13.8
	hiking	18	16.9	19.5	19.7	20	20	19.8	20
	maintenance	16	16	16.1	15.8	11	8	10.7	11.1
	openstacks	0	3.5	0	0.5	0	12.6	0	7
	parking	7	9.7	1.2	4.1	4	7.5	1.4	5.7
	tetris	18	17.1	12.4	14.3	1	5.8	3.2	4.9
	thoughtful	5	5	5	5	8	9	12.7	13.1
	transport	5	3	3.7	4.7	0	0	0	0
	visitall	0	0	0	0	0	0	0	0

		h^{CG} h	h^{CG} hb intra	h^{CG} hB inter	h^{CG} hbB both	h^{CG} hdD both	h^{FF} h	h^{FF} hb intra	h^{FF} hB inter	h^{FF} hbB both	h^{FF} hdD both
	total	187	187.2	206.8	208.7	215.8	192	207.8	232.9	237.7	223.9
IPC11 w/o duplicates	elevators	9	9.2	12.6	13.3	9.7	19	18.2	18.5	19.4	13.7
	nomystery	7	6.4	5.5	5.6	15.1	9	6.6	7.6	6.6	17
	parcprinter	20	19.6	13.7	12.4	18.7	20	20	19.9	18.9	20
	pegsol	20	20	19.7	19.8	20	20	20	20	20	20
	scanalyzer	20	20	20	20	20	15	16.6	19.1	19.1	18.6
	sokoban	16	15.9	15.8	15.2	17	19	18.6	18.5	18.4	17.4
	tidybot	16	17.3	17.5	17.5	18.6	16	15	16.4	16.3	16.7
	woodwork	2	1.8	14	12.8	7.7	2	1.5	14.8	15.7	7.2
IPC14	barman	0	0	0	0	0	0	0	7.6	6.5	1
	cavediving	7	7.1	7	6.9	7	7	7	7	7	7.2
	childsnack	1	0	0.1	0	1.5	0	0	0.1	0	0.3
	citycar	0	0.2	1.1	0.4	4.7	0	0	3	3.8	7.1
	f oortile	0	0	0.5	0.2	2	2	2	2.1	2	2.1
	ged	0	0	4.8	4.6	9.7	19	19.2	12.8	13	13.8
	hiking	18	15.9	18.7	18.8	19.7	20	17.6	19.9	20	20
	maintenance	16	14.6	14.9	14.1	15.8	11	6.7	10	5.8	11.1
	openstacks	0	0.1	2.5	2.4	0.5	0	15.7	11.7	14.5	7
	parking	7	10.4	7.6	10.9	4.1	4	5.4	2.3	4.8	5.7
	tetris	18	19.7	17.6	19.4	14.3	1	8.6	7	11.1	4.9
	thoughtful	5	4.9	5.2	5.2	5	8	9.1	11.2	11	13.1
	transport	5	4.1	6	7.1	4.7	0	0	0	0	0
	visitall	0	0	2	2.1	0	0	0	3.4	3.8	0

Type-based and IP showed different improvements
→ Combining Type-based and IP also improves performance.
Exp4: LAMA + d + D + b + B : LAMAdb^2DB (combining inter/intra Type-based and inter/intra IP)
→ State-of-the-Art performance

LAMA	+d	+D	+dD	+b	+B	+bB	+db^2DB
293.2	296.5	294.3	295.4	293.3	287.6	297.6	**304.5**

6. Future Implication

- Type-GBFS, ε-GBFS, DBFS, IP ... many approaches
- IP shows impressive performance and is based on fractals
- How they differ? And why?

→ Understand them as algorithms creating ***random fractals***
→ Fractals can be **quantitatively** analysed by ***fractal dimension***: Representing how "sparse" the search space is?