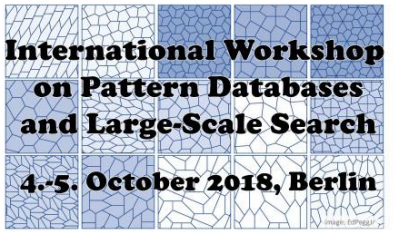
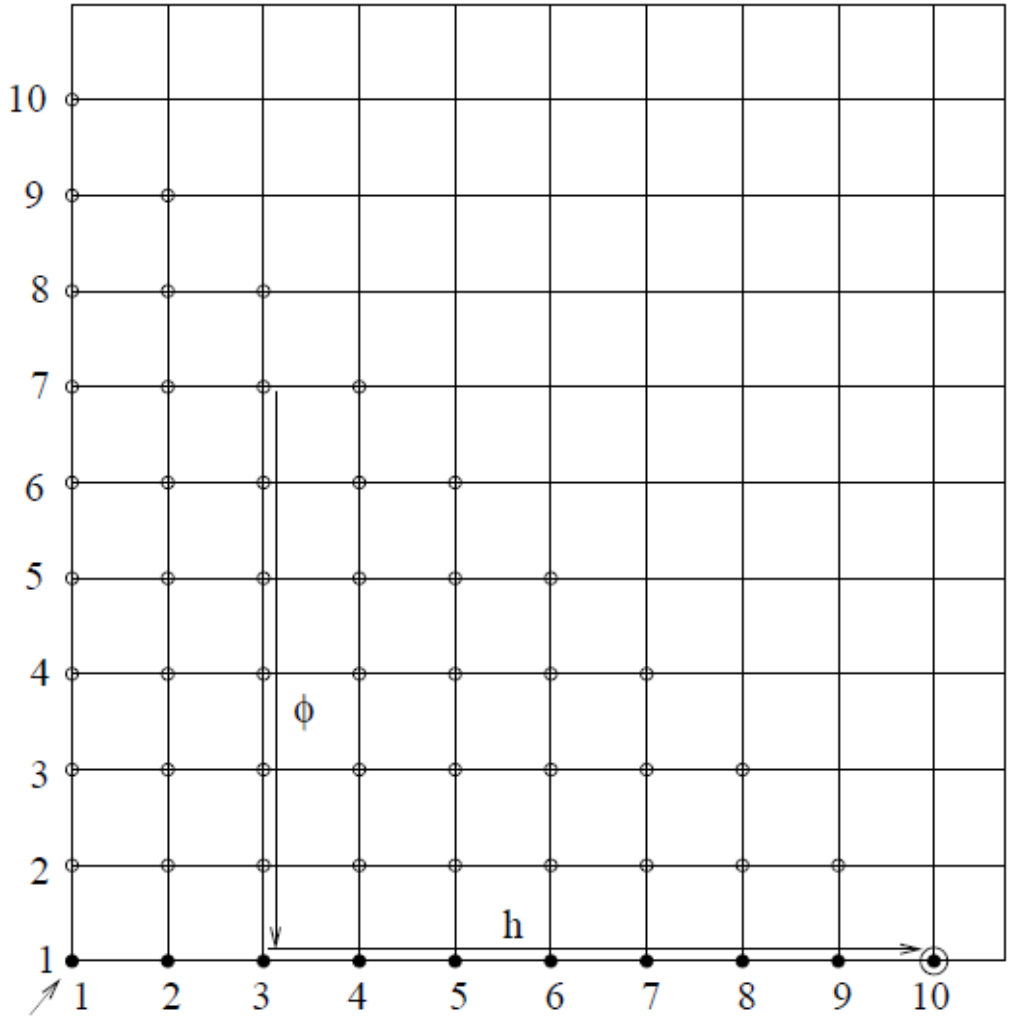
PATTERN DATABASES IN SEARCH AND AI PLANNING

Stefan Edelkamp, King's College London



PROBLEM:

ON-LINE RELAXATION DOES NOT PAY OFF



MARCO VALTORTA: IJCAI-83 "A RESULT ON THE COMPUTATIONAL COMPLEXITY OF HEURISTIC ESTIMATES FOR THE A* ALGORITHM"

Algorithm M: Solve the problem P using A*, where each necessary value of h(n) is computed by solving an auxiliary problem using the Dijkstra algorithm.

Main Theorem.

Let a semantic problem P be given. To solve problem P algorithm M expands at least every node expanded by the Dijkstra algorithm.

Since the Dijkstra algorithm never expands the same node twice, it follows as a corollary that algorithm M uses at least the same number of node expansions as the Dijkstra algorithm.

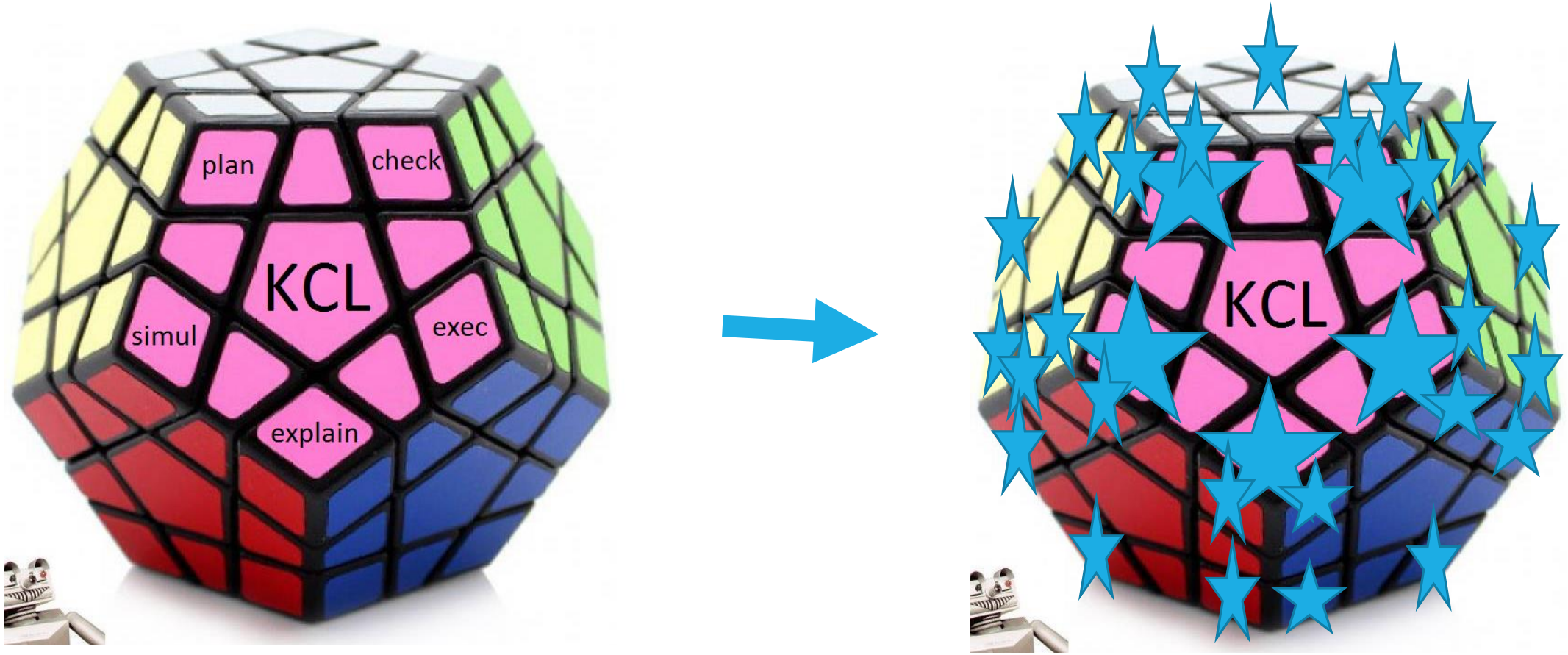
➔ Solution Reuse via Caching, Memorizing

HISTORY → ARIEL'S INVITED TALK

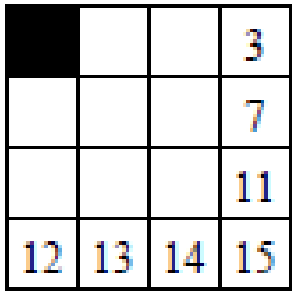
Joseph C. Culberson, Jonathan Schaeffer:
Searching with Pattern Databases. Canadian Conference on AI 1996: 402-416

Joseph C. Culberson, Jonathan Schaeffer:
Pattern Databases. Computational Intelligence 14(3): 318-334 (1998)

Richard E. Korf:
Finding Optimal Solutions to Rubik's Cube Using Pattern Databases. AAAI/IAAI 1997: 700-705


plan
check
KCL
simul
exec
explain
KCL

EFFECT



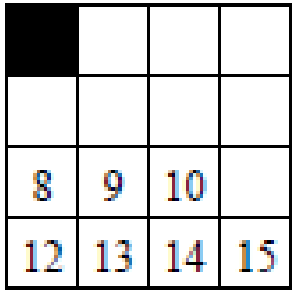
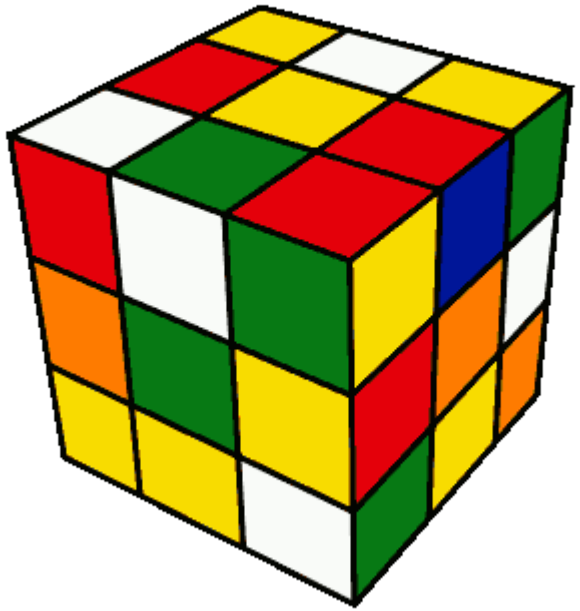


Figure 5.5: Fringe and corner target pattern for the FIFTEEN-PUZZLE.

Heuristic	Nodes	Mean Heuristic Value
Manhattan Distance	401,189,630	36.942
Linear Conflict Heuristic	40,224,625	38.788
5-tile Pattern Database	5,722,922	41.562
6-tile Pattern Database	3,788,680	42.924

Table 5.1: Effect of pattern databases in the FIFTEEN-PUZZLE.

“GOD’S ALGORITHM”



Problem	Depth	Nodes Generated
1	16	3,720,885,493
2	17	11,485,155,726
3	17	64,937,508,623
4	18	126,005,368,381
5	18	262,228,269,081
6	18	344,770,394,346
7	18	502,417,601,953
8	18	562,494,969,937
9	18	626,785,460,346
10	18	1,021,814,815,051

PATTERN DATABASE QUALITY

We assume d to be the average optimal solution length for a random instance and that our pattern database is generated by caching heuristic values for all states up to distance d from the goal. If the abstract search tree is also branching with factor b, a lower bound on the expected value of a pattern database heuristic is $\log_b m$, where m is the number of stored states in the database (being equal to the size of the abstract state space). The derivation of $\log_b m$ is similar to the one of $\log_b n$ for the concrete state space.

R.I.P. HEURISTIC BRANCHING FACTOR

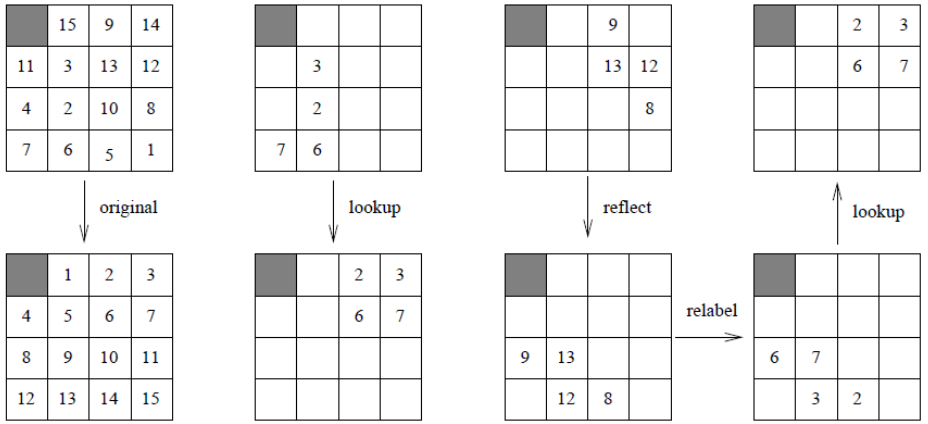
The hope is that the combination of an overly optimistic with a too pessimistic estimate results in a more realistic measure. Let t be the number of nodes generated in an A* search (without duplicate detection). Since d is the depth to which A* must search, we can estimate $d \approx \log_b n$. Moreover, as argued above, we have $\overline{h} \approx \log_b m$, and $t \approx b^{d-\overline{h}}$. Substituting the values for d and $\overline{h}$ yields

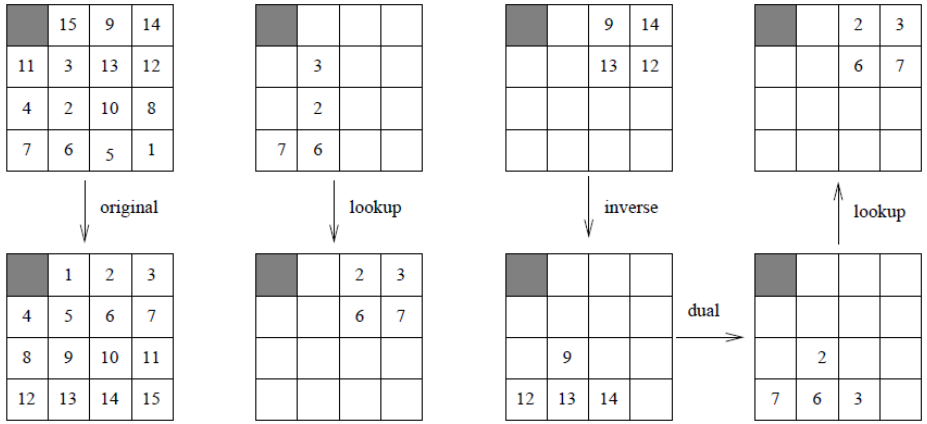
$$t \approx b^{d-\overline{h}} \approx b^{\log_b n - \log_b m} = n/m.$$

IMPACT

Since the treatment is insightful but informal, this estimate has been denoted as *Korf's conjecture*; it states that the number of generated nodes in an A* search without duplicate detection using a pattern database may be approximated by $O(n/m)$, the size of the problem space divided by the available memory. Using experimental data from the RUBIK'S CUBE prolem show that the prediction is very good. We have $n \approx 4.3252 \cdot 10^{19}$, $m = 88,179,940 + 2 \cdot 42,577,920 = 173,335,680$, $n/m = 149,527,409,904$, and $t = 352,656,042,894$, which is off only by a factor of 1.4.

SYMMETRIC AND DUAL DB – TABLE COMPRESSION

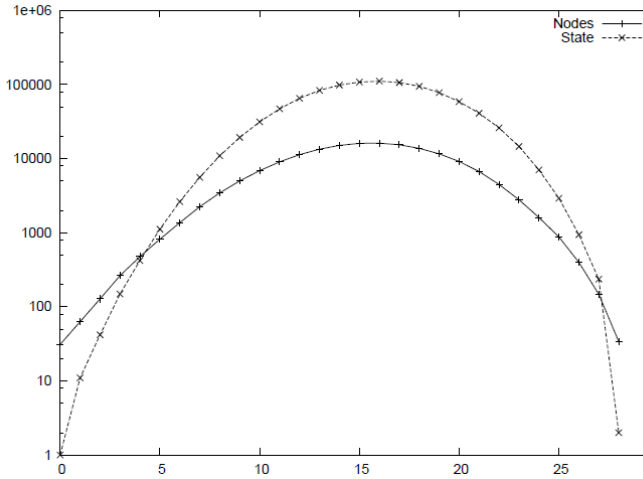


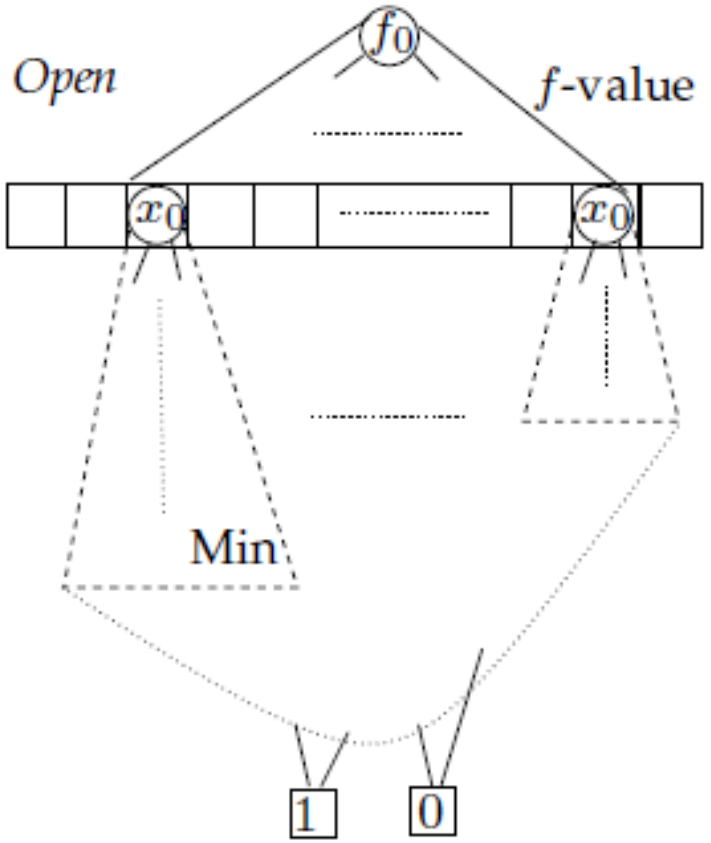


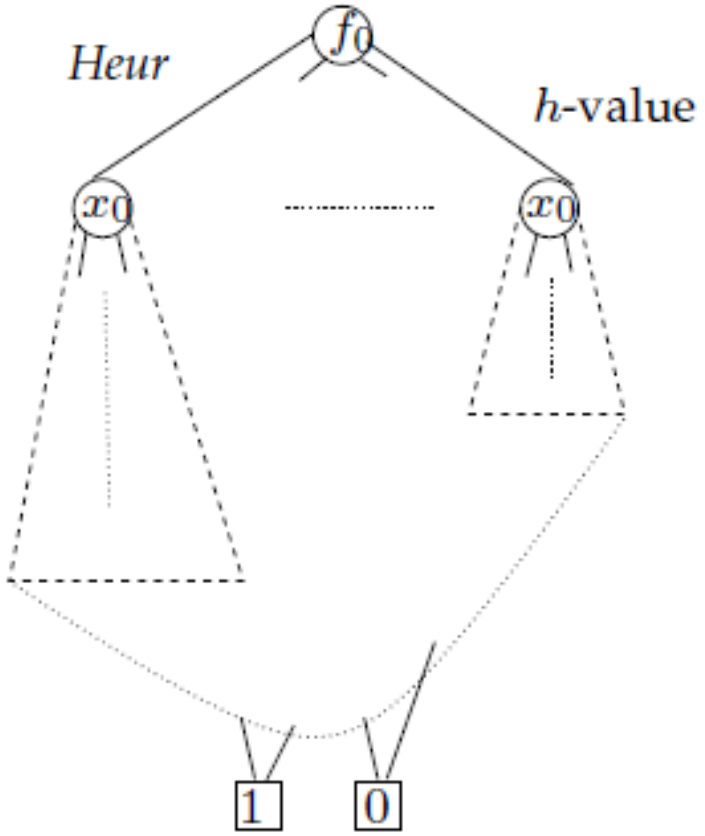
Address	Value
1	4
2	5
3	6
4	7
5	5
6	3
7	2
8	1

Compressed Address (stored)	Original Address (not stored)	Value (stored)
1	{1,2}	4
2	{3,4}	6
3	{5,6}	3
4	{7,8}	1

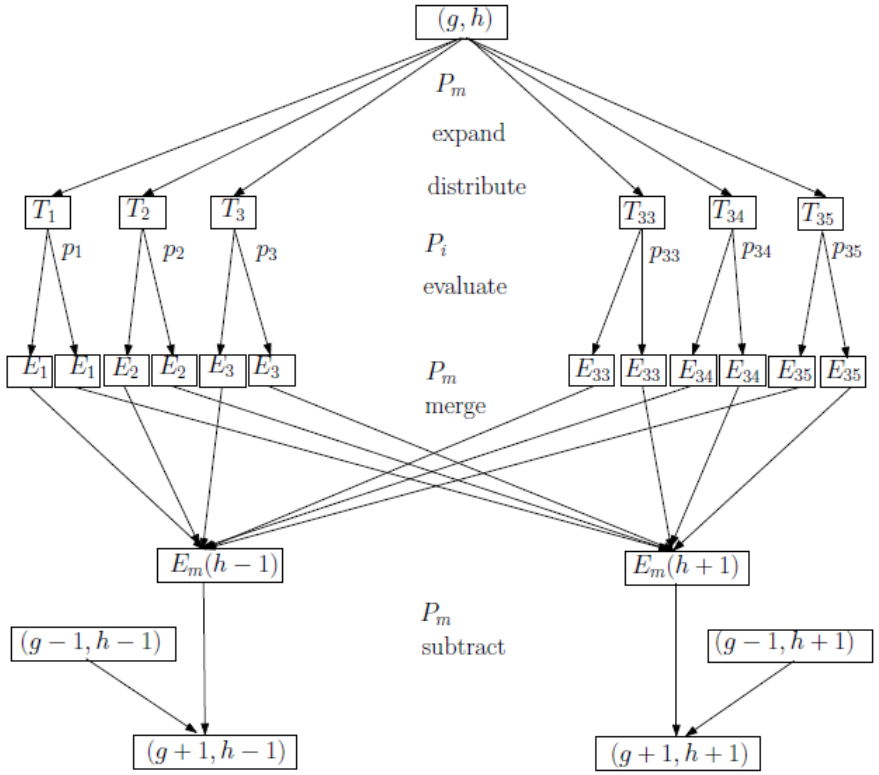
BDD COMPRESSION – SYMBOLIC SEARCH

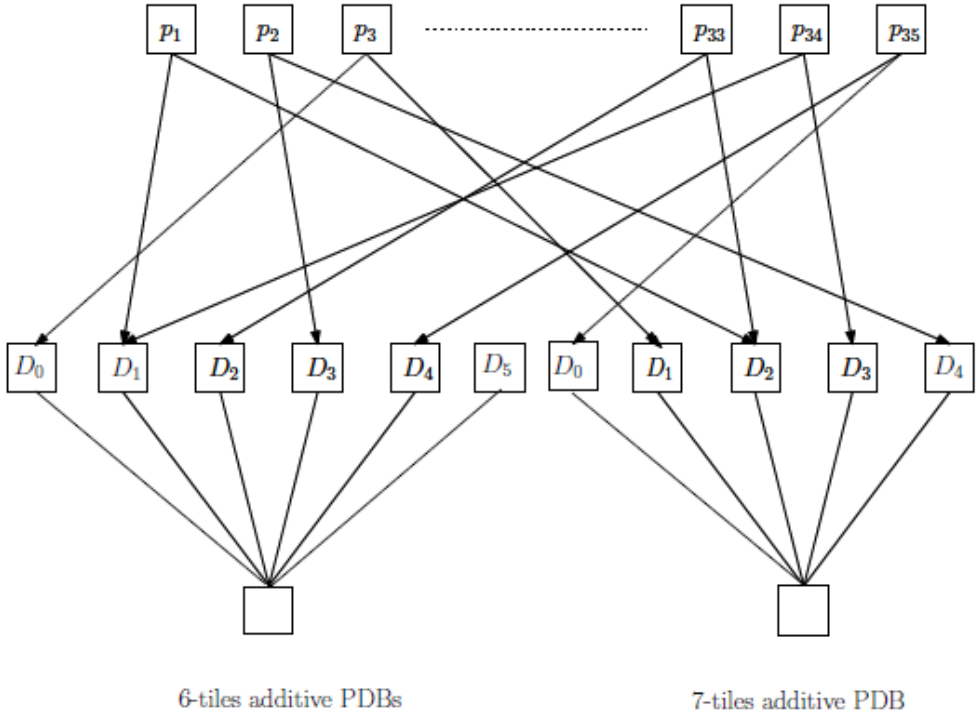






DISTRIBUTED PATTERN DATABASES





6-tiles additive PDBs

7-tiles additive PDB

GENERAL GRAPH PATTERN DATABASES

Graph G=(V,E), possibly directed, weighted, start s, goal t

Abstraction mapping a: V → V', E → E' (e.g., supernode contraction, additional edes)

such that each path in G also present in G=(V',E',s',t'), with shorter cost

May introduce spurious paths

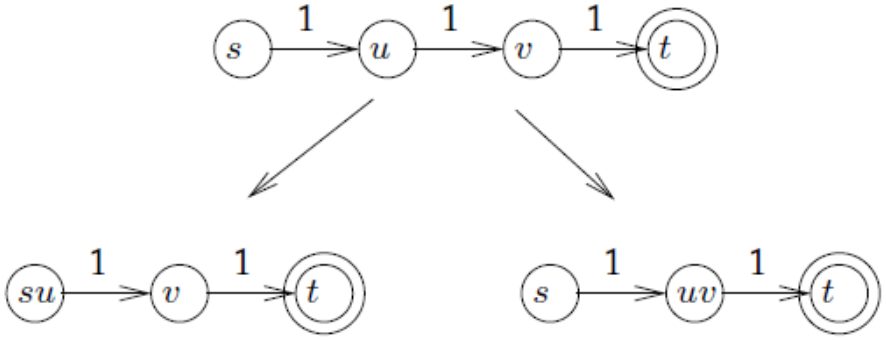
(simulation, behaviour)

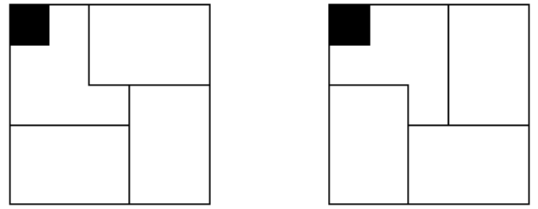
Shortest path in G' lower bound on one in G

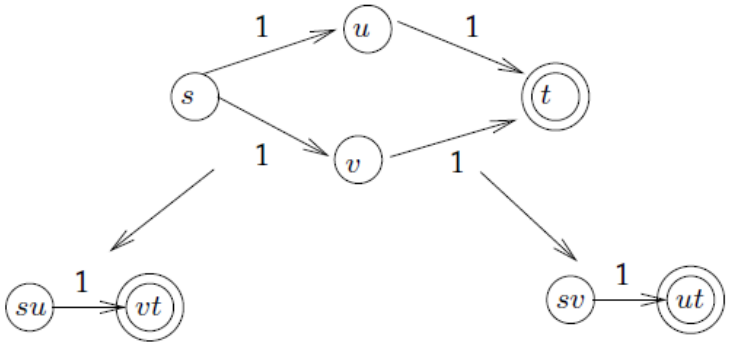
Triangle inequality ➔ consistency

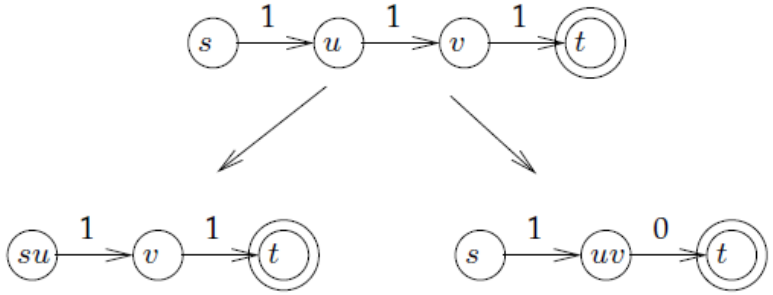
Pattern Database: storing results of Dijkstra SSSP Algorithm Inv-G in Table

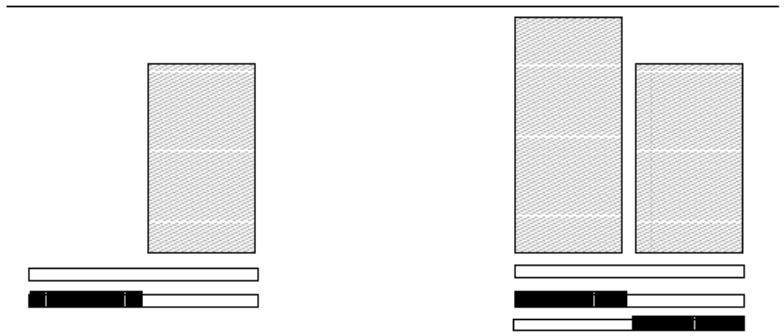
MULTIPLE PATTERN DB





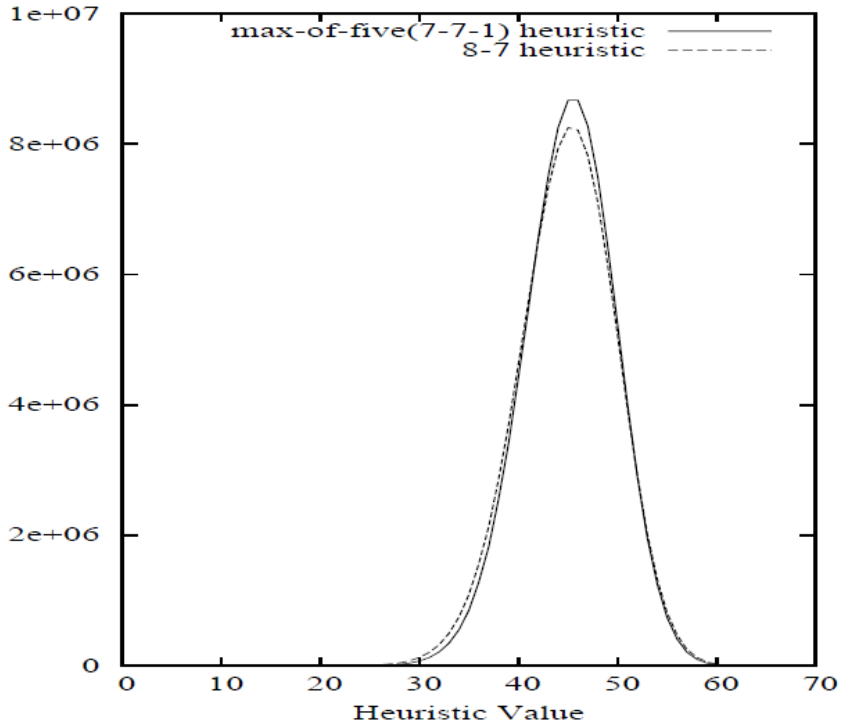






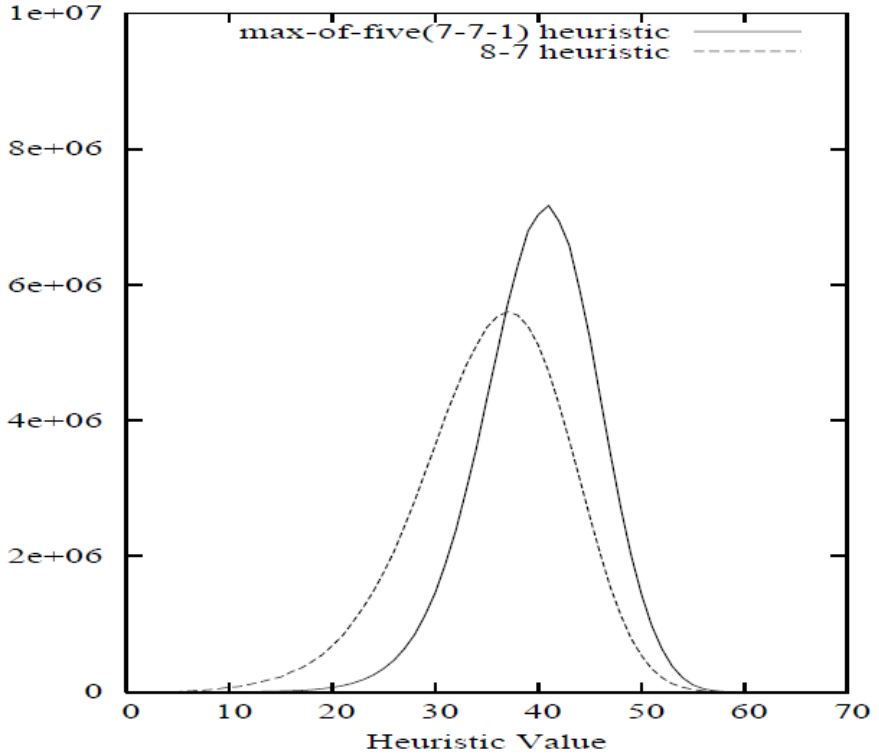
CHICKEN AND OXES

STATIC



Proposition 1: The use of smaller pattern databases instead of one large pattern database usually reduces the number of patterns with high h-values, and often reduces the maximum achievable h-value, but the max'ing of the smaller pattern databases can make the number of patterns with low h-values significantly smaller than the number of low-valued patterns in the larger pattern database.

RUNTIME



Proposition 2: Eliminating low h-values is more important for improving search performance than retaining large h-values.

This assertion is not immediately obvious. To see why it is true, consider the formula developed in (Korf, Reid, & Edelkamp 2001) to predict the number of nodes generated by one iteration of IDA* to depth d:

$$\sum_{i=0}^{d} N(i) \cdot P(d-i) \qquad (1)$$

STRIPS PLANNING

Set of Atoms/Facts F = {f1,….,fn}

Initial State I, Goal Condition G:

Conjunctions of F

Operators: (pre,add,del) with

pre, add, del Conjunction of F

Lifted/Grounded Representation

coffee_errors.pddl

```
1  (define COFFEE
2
3    (requirements
4      :typing)
5
6    (:types room - location
7            robot human _ agent
8            furniture door - (at ?l - location)
9            kettle ?coffee cup water - movable
10           location agent movable - object)
11
12   (:predicates (at ?l - location ??o - object)
13                (have ?m - movable ?a - agent)
14                (hot ?m - movable) = true
15                (on ?f - furniture ?m - movable))
16
17   (:action boil
18     :parameters (?m - movable $k - kettle ?a - agent)
19     :preconditions (have ?m ?a)
20     :effect (hot ?m))
21
```

Line 20, Column 22 Spaces: 2 PDDL

PROJECTION STRIPS PLANNING

Set of Atoms/Facts a(F) = {a(f1),….,a(fn)}

Projection: a(f) either f or {}

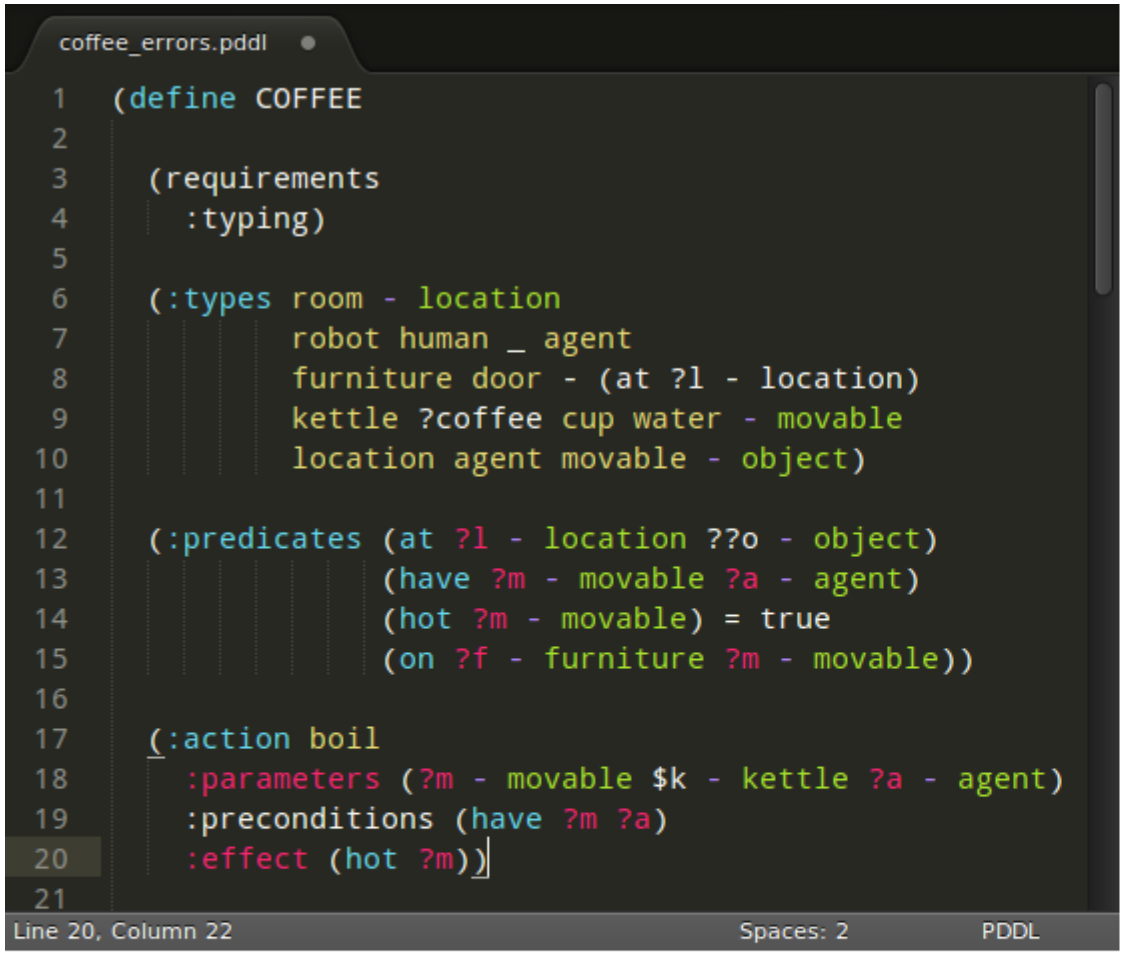
Initial State a(I), Goal Condition a(G):

Conjunctions of a(F)

Operators: (a(pre),a(add),a(del)) with

a(pre), a(add), a(del) Conjunctions of a(F)

Extends to Finite Domain Variables, SAS+

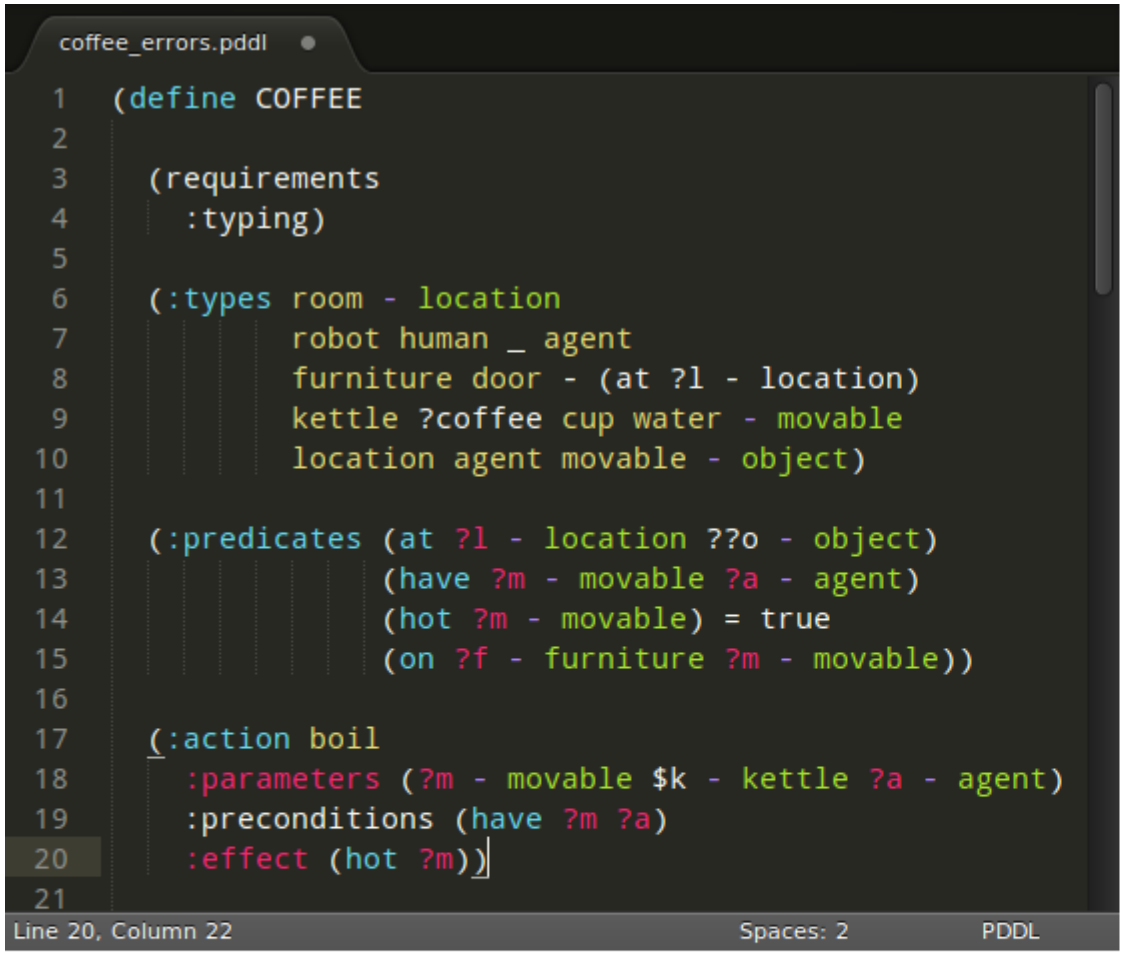


ABSTRACT STRIPS PLANNING

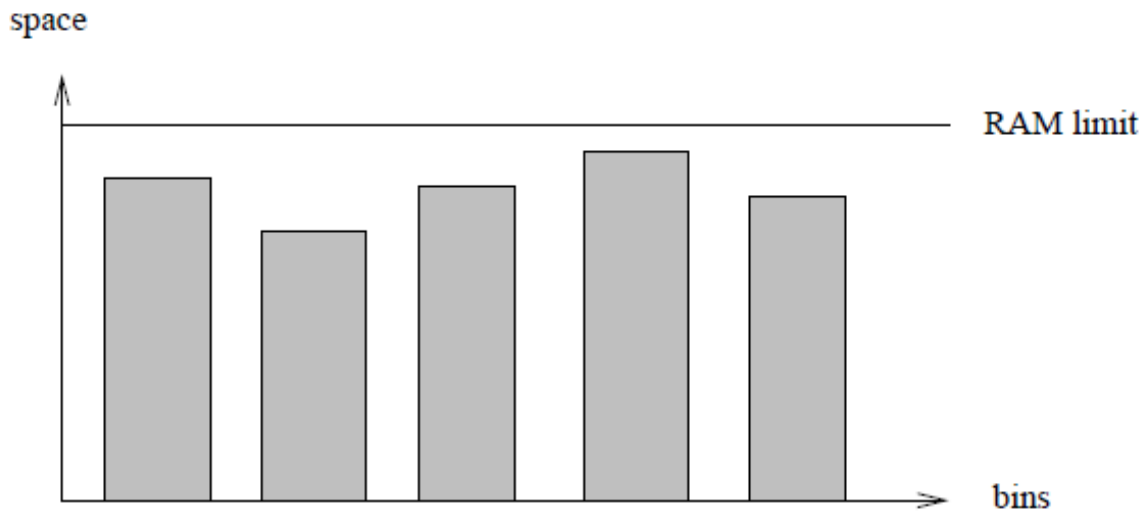
Initial State a(I), Goal Condition a(G)

Operators: (a(o))

Label Set Reduction…

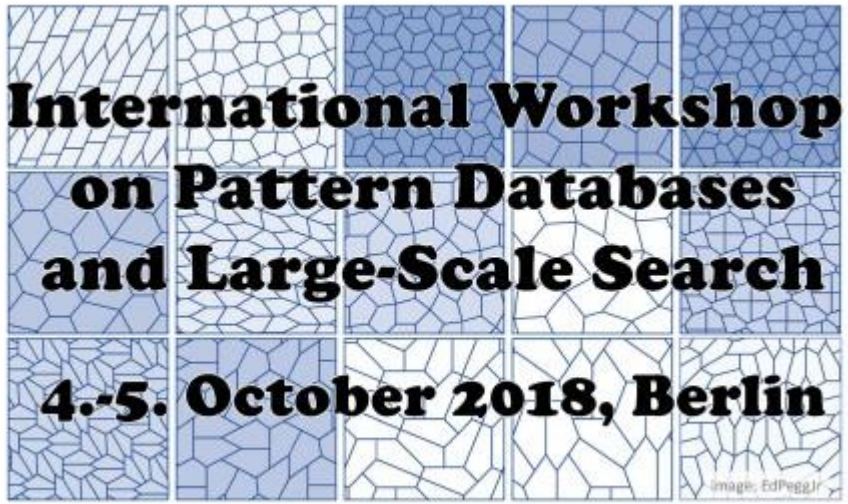


BIN PACKING FOR AUTOMATED PATTERN SELECTION



- * INSTEAD OF +
- GAS TO THE RESCUE

QUESTION:
WHAT IS A PATTERN?



REFERENCES

Stefan Edelkamp:
Symbolic Pattern Databases in Heuristic Search Planning. AIPS 2002: 274-283

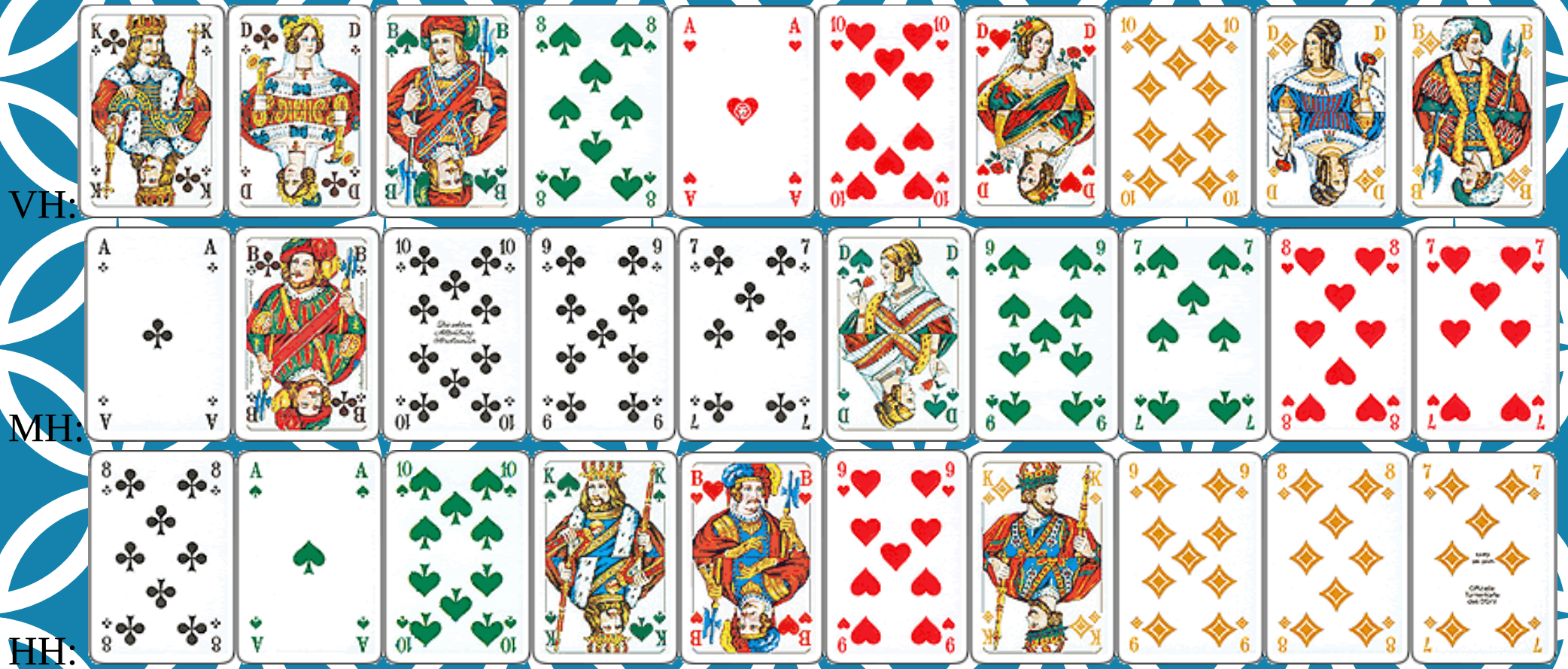
Stefan Edelkamp, Shahid Jabbar, Peter Kissmann:
Scaling Search with Pattern Databases. MoChArt 2008: 49-64

Stefan Edelkamp, Peter Kissmann, Álvaro Torralba:
Symbolic A* Search with Pattern Databases and the Merge-and-Shrink Abstraction. ECAI 2012: 306-311

Stefan Edelkamp, Peter Kissmann:
Partial Symbolic Pattern Databases for Optimal Sequential Planning. KI 2008: 193-200

Stefan Edelkamp:
Automated Creation of Pattern Database Search Heuristics. MoChArt 2006: 35-50

Stefan Edelkamp:
External Symbolic Heuristic Search with Pattern Databases. ICAPS 2005: 51-60



PLAYING WORLD-CLASS COMPUTER SKAT

Stefan Edelkamp and Rainer Goessl

MORE TO ADD...