Diversifying Greedy Best-First Search by Clustering States

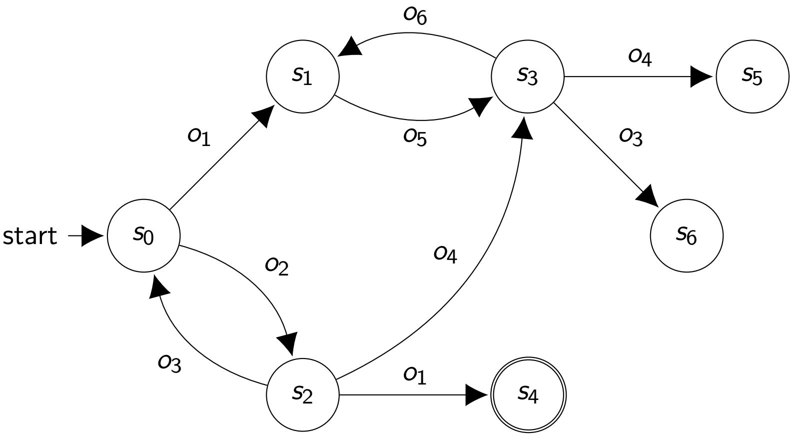
Daniel Killenberger

Department of Mathematics and Computer Science
Natural Science Faculty of the University of Basel

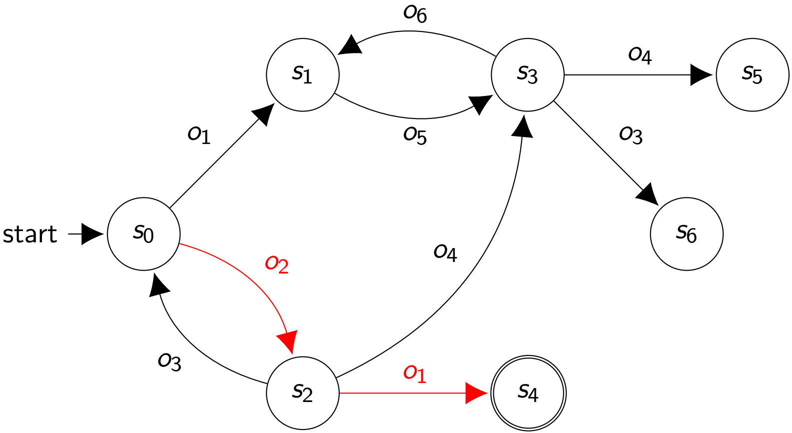
March 16th 2017

Introduction

Introduction - Classical Planning

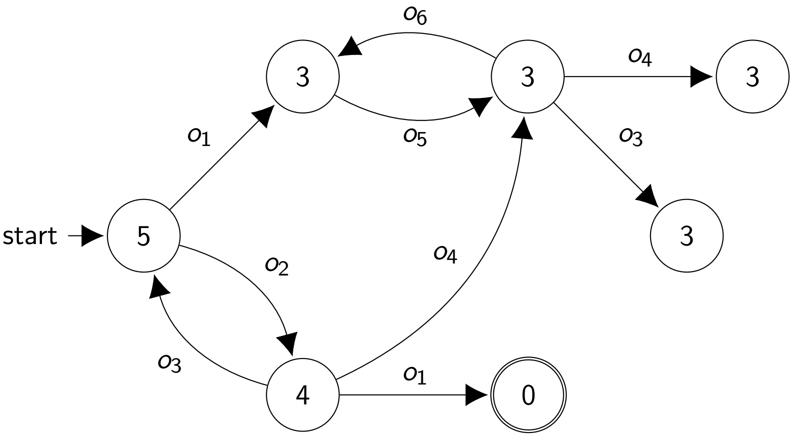


Introduction - Classical Planning

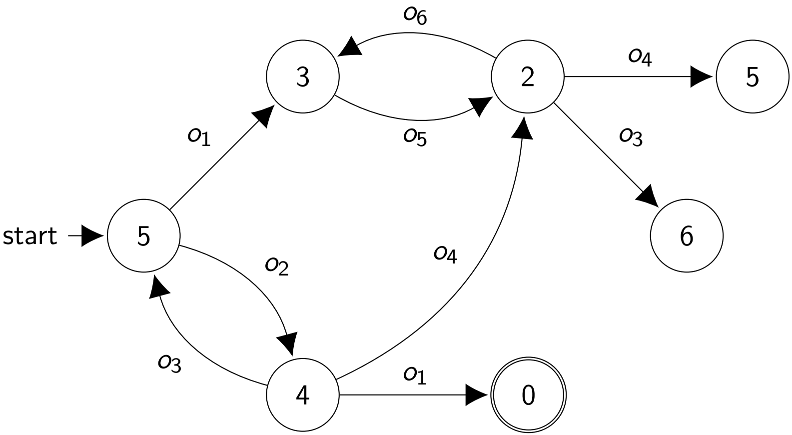


$\pi =< o_2, o_1 >$

Greedy Best-First Search - Plateau



Greedy Best-First Search - Local Minima



Cluster-based Open List

Cluster-based Open List

- cluster based on *similarity*
- k amount of clusters
- optional reclustering using k-means algorithm
- state retrieval by returning a random state from a random cluster

SAS+ Formalism

$V = \{v_1, v_2\}$

$dom(v_1) = 2,\ dom(v_2) = 3$

SAS+ Formalism

$V = \{v_1, v_2\}$

$dom(v_1) = 2, \ dom(v_2) = 3$

$S = \{\langle 0, 0\rangle, \ \langle 1, 0\rangle, \ \langle 1, 1\rangle, \ \langle 1, 2\rangle, \ \langle 0, 1\rangle, \ \langle 0, 2\rangle\}$

Similarity

$V = \{v_1, v_2\}$

$dom(v_1) = 2, \; dom(v_2) = 3$

$s_1 = \langle 1, 1 \rangle$
$s_2 = \langle 0, 2 \rangle$
$s_3 = \langle 1, 0 \rangle$

Hamming distance

$s_1 = \langle 1, 1 \rangle$
$s_2 = \langle 0, 2 \rangle$

$d(s_1, s_2) = 2$

Hamming distance

$s_1 = \langle 1, 1 \rangle$
$s_3 = \langle 1, 0 \rangle$

$d(s_1, s_3) = 1$

Clustering

$s_1 = \langle 1, 1 \rangle$
$s_2 = \langle 0, 2 \rangle$
$s_3 = \langle 1, 0 \rangle$

Amount of clusters $k = 2$
$c_1 = \{s_1, s_3\}$
$c_2 = \{s_2\}$

Binary Representation

$dom(v) = 3$ and $s[v] = 1 \Rightarrow 010$

Binary Representation

$$s_1 = \langle 1, 1 \rangle \Rightarrow bin(s_1) = 01010$$
$$s_2 = \langle 0, 2 \rangle \Rightarrow bin(s_2) = 10001$$
$$s_3 = \langle 1, 0 \rangle \Rightarrow bin(s_3) = 01100$$

Cluster Average

$$s_1 = \langle 1, 1 \rangle \Rightarrow bin(s_1) = 01010$$
$$s_2 = \langle 0, 2 \rangle \Rightarrow bin(s_2) = 10001$$
$$s_3 = \langle 1, 0 \rangle \Rightarrow bin(s_3) = 01100$$

$$c_1 = \{s_1, s_3\}$$
$$c_2 = \{s_2\}$$

Cluster Average

$$s_1 = \langle 1, 1 \rangle \Rightarrow bin(s_1) = 01010$$
$$s_2 = \langle 0, 2 \rangle \Rightarrow bin(s_2) = 10001$$
$$s_3 = \langle 1, 0 \rangle \Rightarrow bin(s_3) = 01100$$

$$c_1 = \{s_1, s_3\}$$
$$c_2 = \{s_2\}$$

$$c_1.factsMean = \langle 0, 1, \tfrac{1}{2}, \tfrac{1}{2}, 0 \rangle$$
$$c_2.factsMean = \langle 1, 0, 0, 0, 1 \rangle$$

Distance Function

$$d(s,c) = \sum_{i=0}^{N-1} |factsMean[i] - bin(s)[i]| \tag{1}$$

State Insertion

$$s_4 = \langle 0, 0 \rangle \Rightarrow bin(s_4) = 10100$$

State Insertion

$$\begin{aligned} bin(s_4) &= \langle 1, 0, 1, 0, 0 \rangle \\ c_1.factsMean &= \langle 0, 1, \tfrac{1}{2}, \tfrac{1}{2}, 0 \rangle \end{aligned}$$

$$d(s_4, c_1) = 3$$

State Insertion

$$bin(s_4) = \langle 1, 0, 1, 0, 0 \rangle$$
$$c_2.factsMean = \langle 1, 0, 0, 0, 1 \rangle$$

$$d(s_4, c_2) = 2$$

Insert s_4 into c_2

State Insertion

While there are empty clusters, insert states into one of those empty clusters.

State insertion

Calculating k distances for every generated state is very computationally demanding $\Rightarrow$ inefficient

K-means Reclustering

Standard K-means Clustering

1. Initialize: Generate k random mean vectors.
2. Cluster: Associate every data point with the closest mean.
3. Centroid: Calculate the centroid of every cluster to get the new representative mean vectors.
4. Convergence: Check for convergence. If criteria not yet fulfilled jump to step 2.

Adjusted K-means Reclustering

1. Cluster: Associate every data point with the closest mean.
2. Centroid: Calculate the centroid of every cluster to get the new representative mean vectors.
3. Time Limit: Check if time limit has been reached. If not jump to step 1.

May produce empty clusters!

Evaluation

Experiments

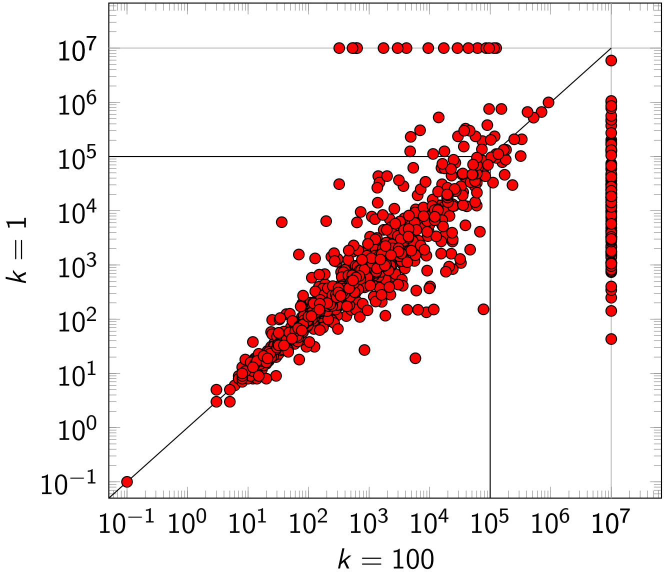
- FF-heuristic as the GBFS evaluator function
- Alternating open list
  - Inserts states into all open lists
  - Picks open list to retrieve state from in round-robin manner
- Three minute time limit
- 2GB Memory Limit

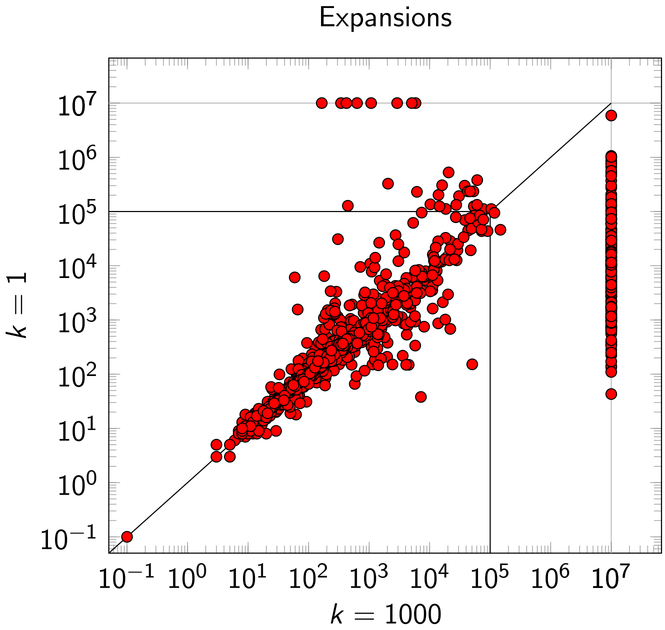
Comparing Amount of Clusters

k	1	10	100	1000	10000	100000
Coverage - Sum	**1726**	1674	1615	1468	1373	1527

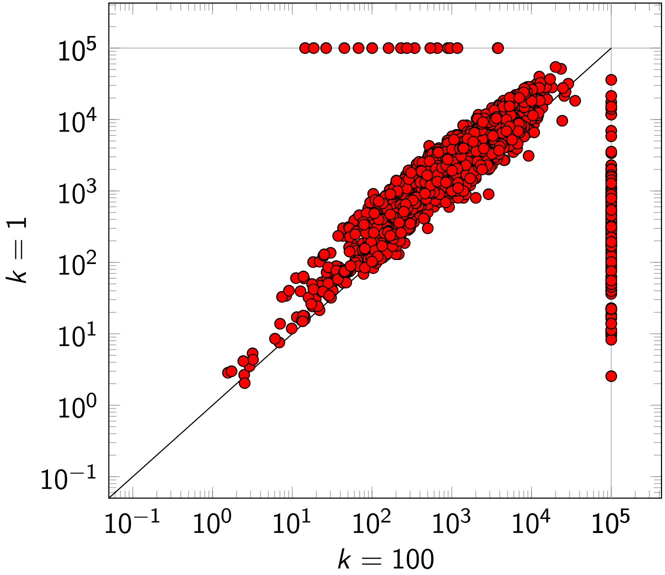
Comparison between different amounts of clusters - coverage of 2532 satisficing planning tasks

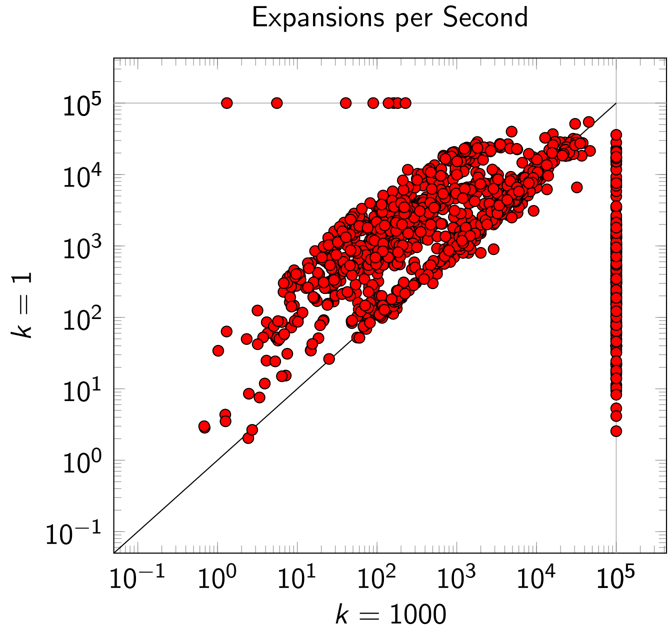
Expansions




Expansions
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
10^{6}
10^{7}
$k = 1$
$k = 1000$

Expansions per Second




Expansions per Second
10^5
10^4
10^3
10^2
10^1
10^0
10^{-1}
$k = 1$
10^{-1} 10^0 10^1 10^2 10^3 10^4 10^5
$k = 1000$

Reclustering

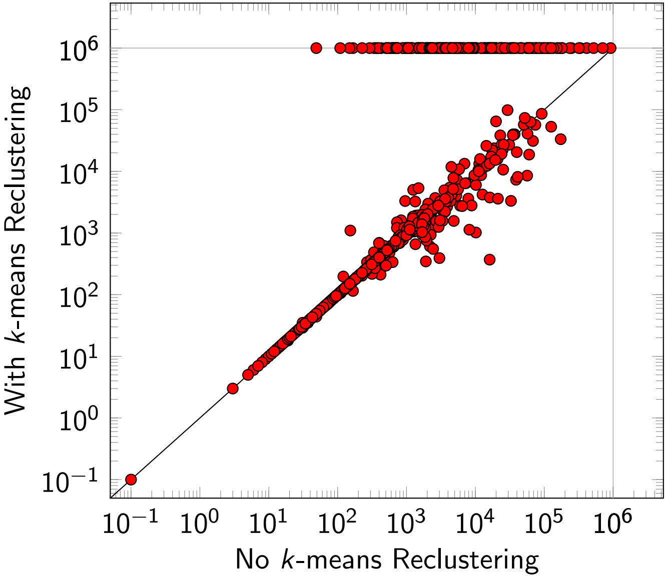
- time limit $l = 2$
- time interval $i = 0.2$

Reclustering vs no Reclustering

	No Reclustering	With Reclustering
Coverage - Sum	**1615**	1372

Comparison between reclustering and not reclustering with 100 clusters - coverage of 2532 satisficing planning tasks

Expansions


10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
10^{6}
With k-means Reclustering
No k-means Reclustering

Conclusion

Conclusion

- Amount of expansions reduced for hard planning tasks compared to random exploration
- Inefficient implementation
- General trend of reduced amount of expansions when comparing reclustering vs not reclustering