Easily Searched Encodings for Number Partitioning

Stuart M. Shieber
Division of Applied Sciences
Harvard University

(joint work with Wheeler Ruml, Tom Ngo, and Joe Marks)

Number Partitioning

Instance: A sequence $A = (a_1, \ldots, a_n)$ of positive rational numbers.

Problem: Find a corresponding sequence $S = (s_1, \ldots, S_n)$ of signs (-1 or $+1$) such that the *residue* u is minimized.

$$u \equiv \left| \sum_{i=1}^{n} s_i a_i \right|$$

6	5	4	4	3	u	Remark
+	+	−	−	−	0	Optimal solution
+	−	+	−	−	2	KK solution

Number Partitioning (continued)

- The Number Partitioning problem:
  - is NP-hard
  - poses problems for local optimization
  - has an extremely effective deterministic heuristic, the *Karmarkar-Karp algorithm* (KK)
- Applications in scheduling, perhaps cryptography

The Conventional Wisdom

- Johnson et al.:

  > [T]raditional local optimization algorithms are not competitive with other techniques for [Number Partitioning], in particular the "differencing" algorithm of Karmarkar and Karp. Consequently, it seems unlikely that simulated annealing, which in essence is a method for improving local optimization, can offer enough of an improvement to bridge the gap.

- Karmarkar et al.:

  > [The KK algorithm] was a great improvement over other results ... It would be very interesting, though quite possibly very difficult, to improve upon that algorithm.

Conclusions

- Stochastic optimization can be successfully applied to number partitioning.
- Problem representation is crucial; search method is essentially unimportant.
- General principles can be adduced.

The KK Algorithm

- **for** $i = 1$ **to** $|A| - 1$

  $j \leftarrow$ the index of the largest element of A

  $k \leftarrow$ the index of the

  second largest element of A

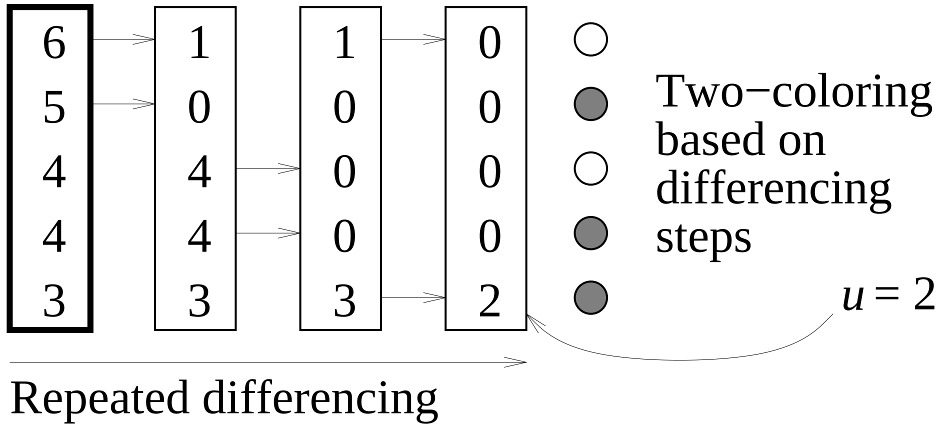
  $a_j \leftarrow a_j - a_k$

  $a_k \leftarrow 0$

  end for

- 2-color based on differencing steps to get a partition

- KK expected difference is $O(\frac{1}{n^{\alpha \log n}})$, but expected optimum is $O(\frac{\sqrt{n}}{2^n})$



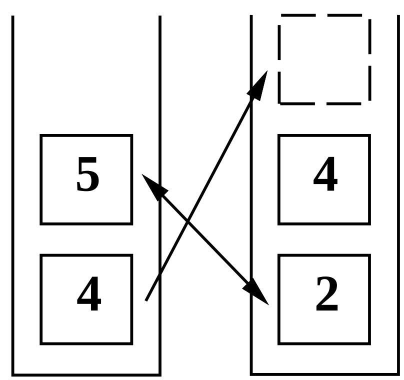
Stochastic Optimization

Search a space of **representations** of solutions by stochastically generating candidate solutions.

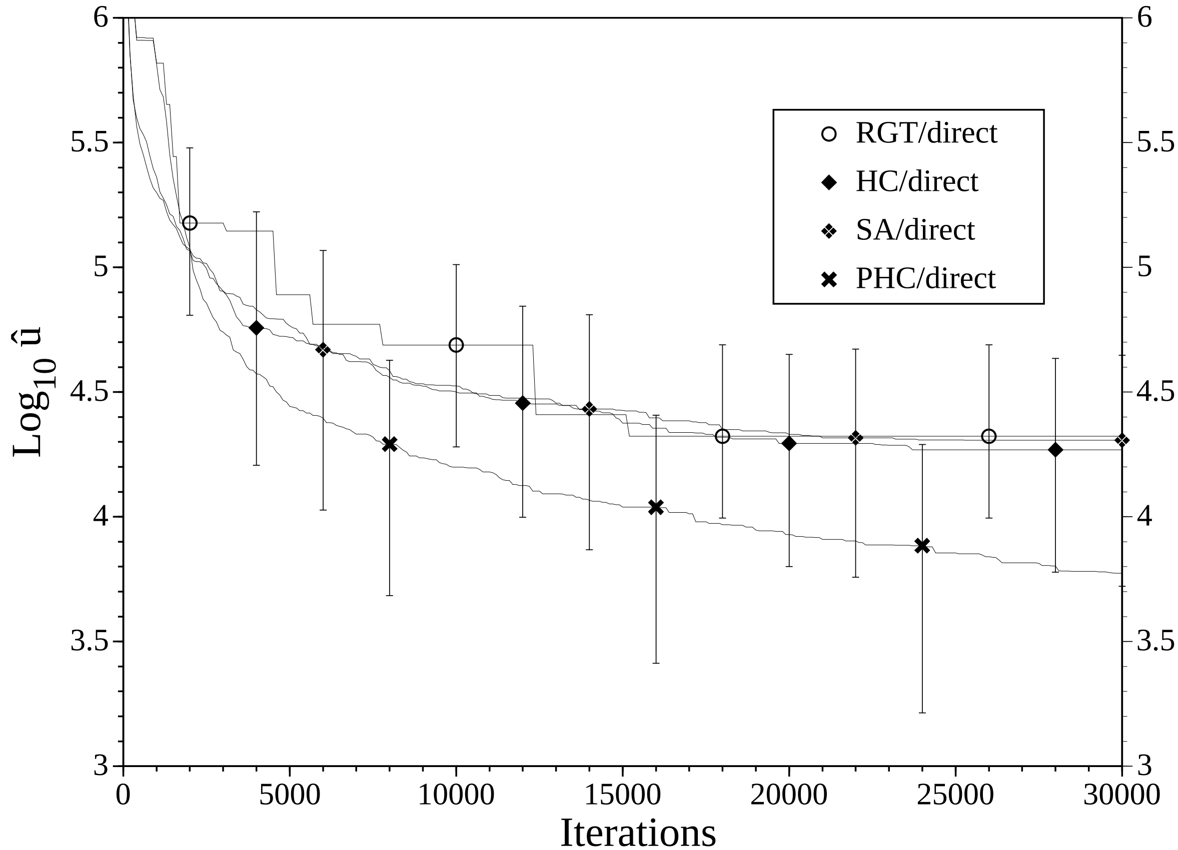
- Random generate-and-test (RGT)
- Hill climbing (HC)
- Simulated annealing (SA)
- Parallel hill climbing (PHC)
- Genetic algorithm (GA)

The Direct Representation

- The representation of the solution — a sign sequence S — is the sign sequence itself.
- The operators for generating solutions are
  - moving
  - swapping



The Direct Representation: Performance



$$\hat{u} = u/u_{KK}$$

Parallel Hill Climbing

- Initialize a population P of random candidate solutions
- For some number of iterations do:
  - Randomly select an operator o
  - Randomly select a solution $s \in P$ (Better solutions are selected with higher probability.)
  - Add $s' = o(s)$ to P
  - Rank solutions in P andelete the lowest-ranking solution
- Return the best solution in P

How to Generate a Good Representation

1. Select an existing greedy heuristic for the optimization problem in question.

2. Modify the heuristic so that its operation is not fully determined.

3. Choose, for every underdeterminedecision point in the heuristic, a parameter whose value wildetermine the decision.

The set of such parameters will be the independent variables of the encoding.

How to Parameterize a Greedy Heuristic

Parameterized arbitration: Representation selects among *arbitrary decisions* made by heuristic.

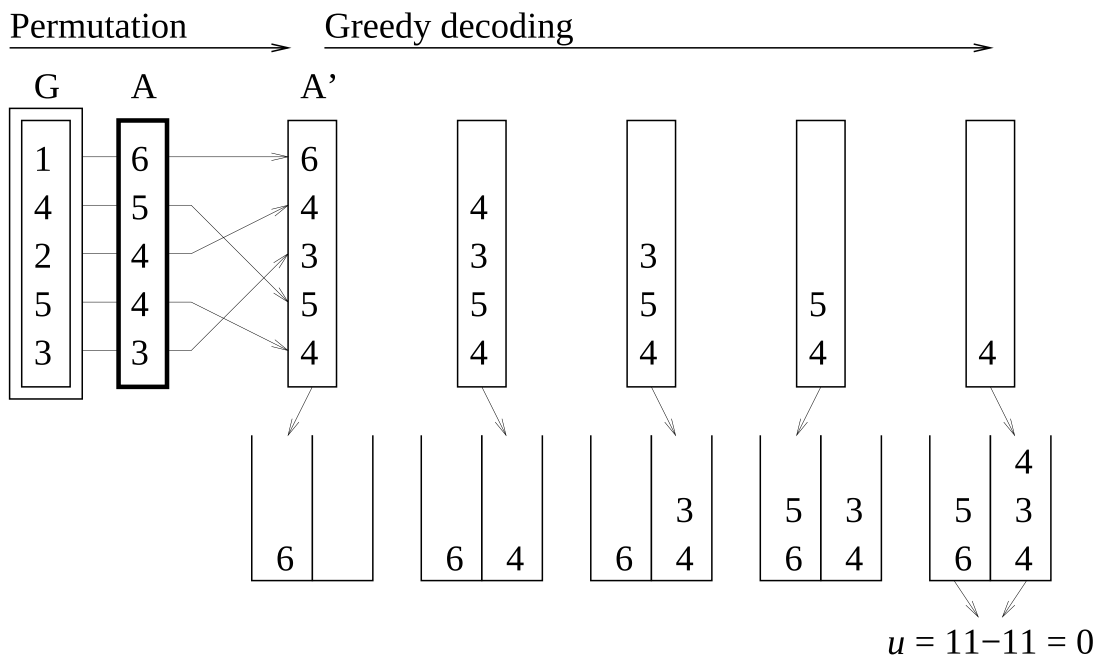
Parameterized constraint: Representation specifies *external constraints* respected by the heuristic.

Parameterized greediness: Representation specifies *degree of greediness* of each decision.

The Greedy Decoder Representation

- A candidate solution is represented by a permutation G of the numbers in A. The solution generation operator is random swapping.

- The sign sequence is generated by a greedy decoder (parameterized arbitration):


Permutation
Greedy decoding
G
A
A'
1 4 2 5 3
6 5 4 4 3
6 4 3 5 4
4 3 5 4
3 5 4
5 4
4
6
6 4
6 3 4
5 6 3 4
5 6 4 3 4
u = 11−11 = 0

Parameterizing the KK Algorithm

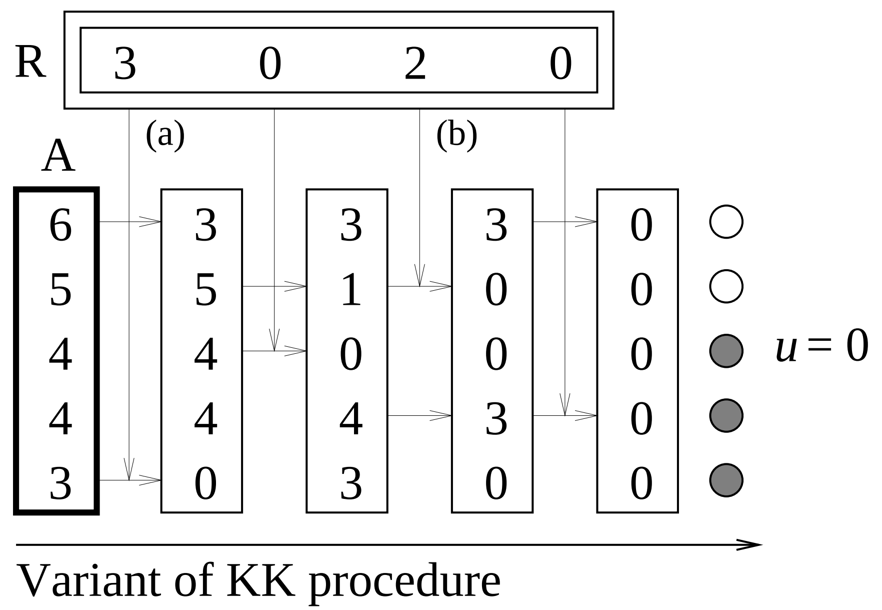
Index-based differencing: Choose the largest and r_i-th largest element to difference at each iteration i. (*parameterized greediness*)

Prepartitioning: Let KK operate over predefined partitions of elements. (*parameterized constraint*)

Index-Based Differencing

- A candidate solution is represented by a list R of indices. The solution generation operator is random replacement of the elements of R.

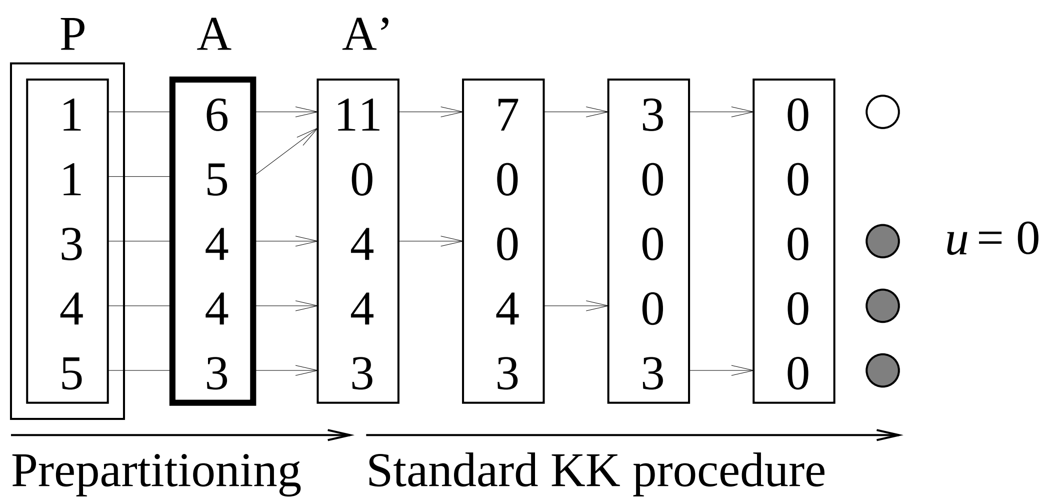
- A KK decoder is used to generate the sign sequence from R:



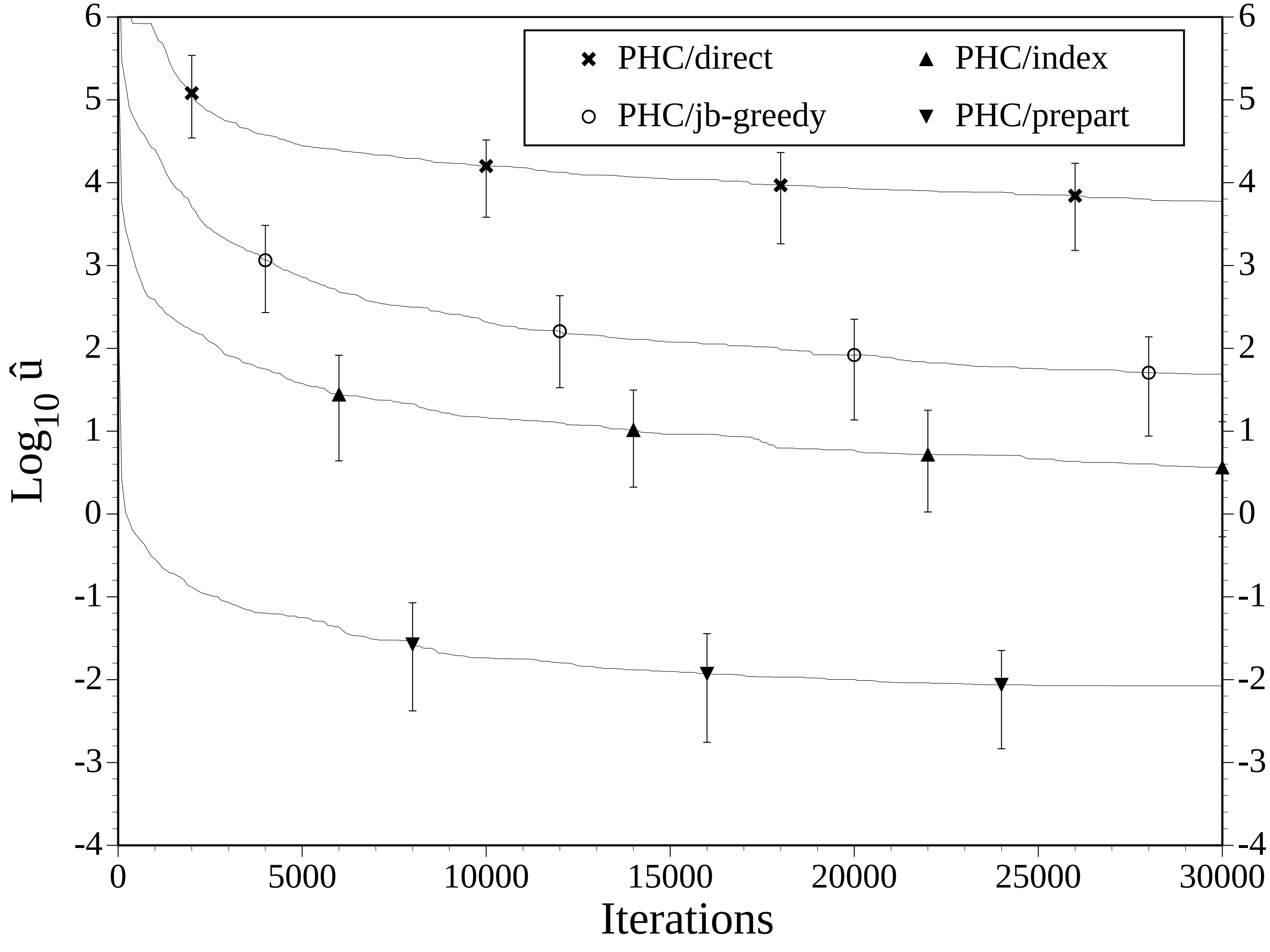
The Prepartitioning Representation

- A candidate solution is represented by a partition P of the elements, an assignment to each element of a partition number. The solution generation operator is random perturbation of the partition numbers in P.

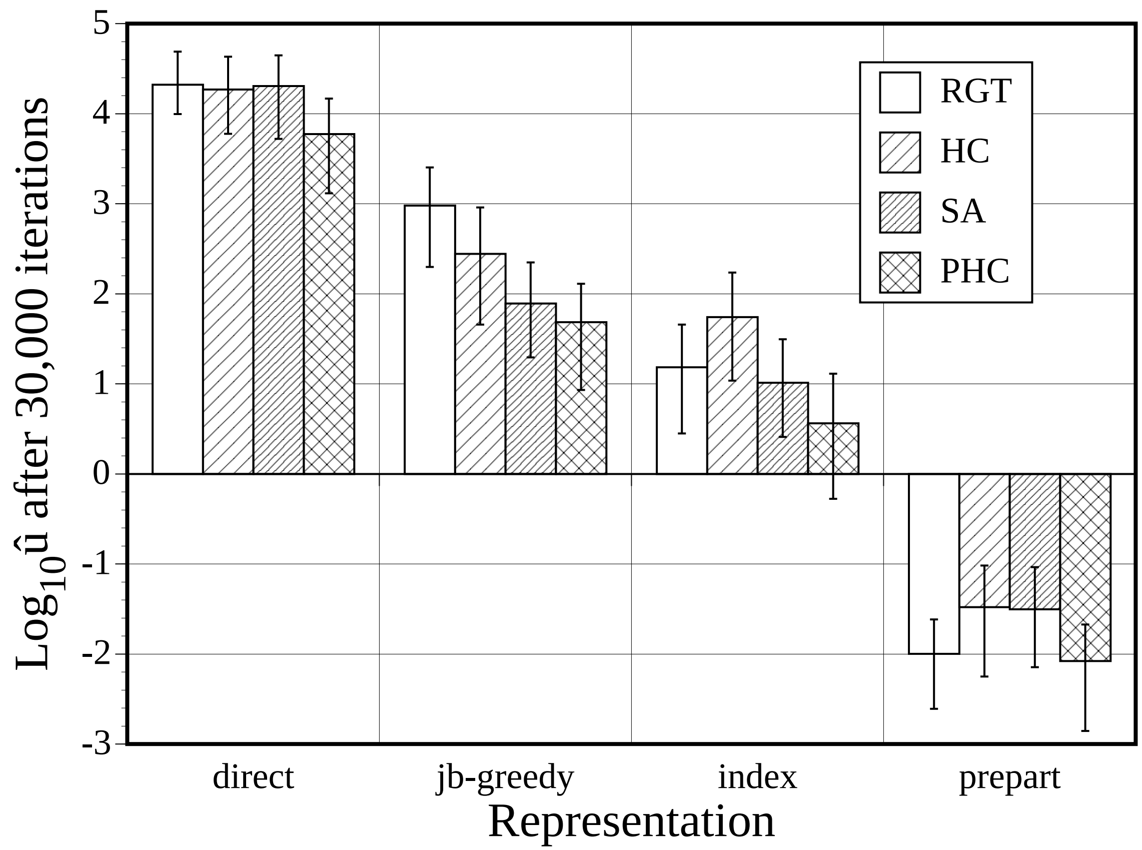
- The KK algorithm is used to generate the sign sequence from P:



Performance of the Representations (+ PHC)

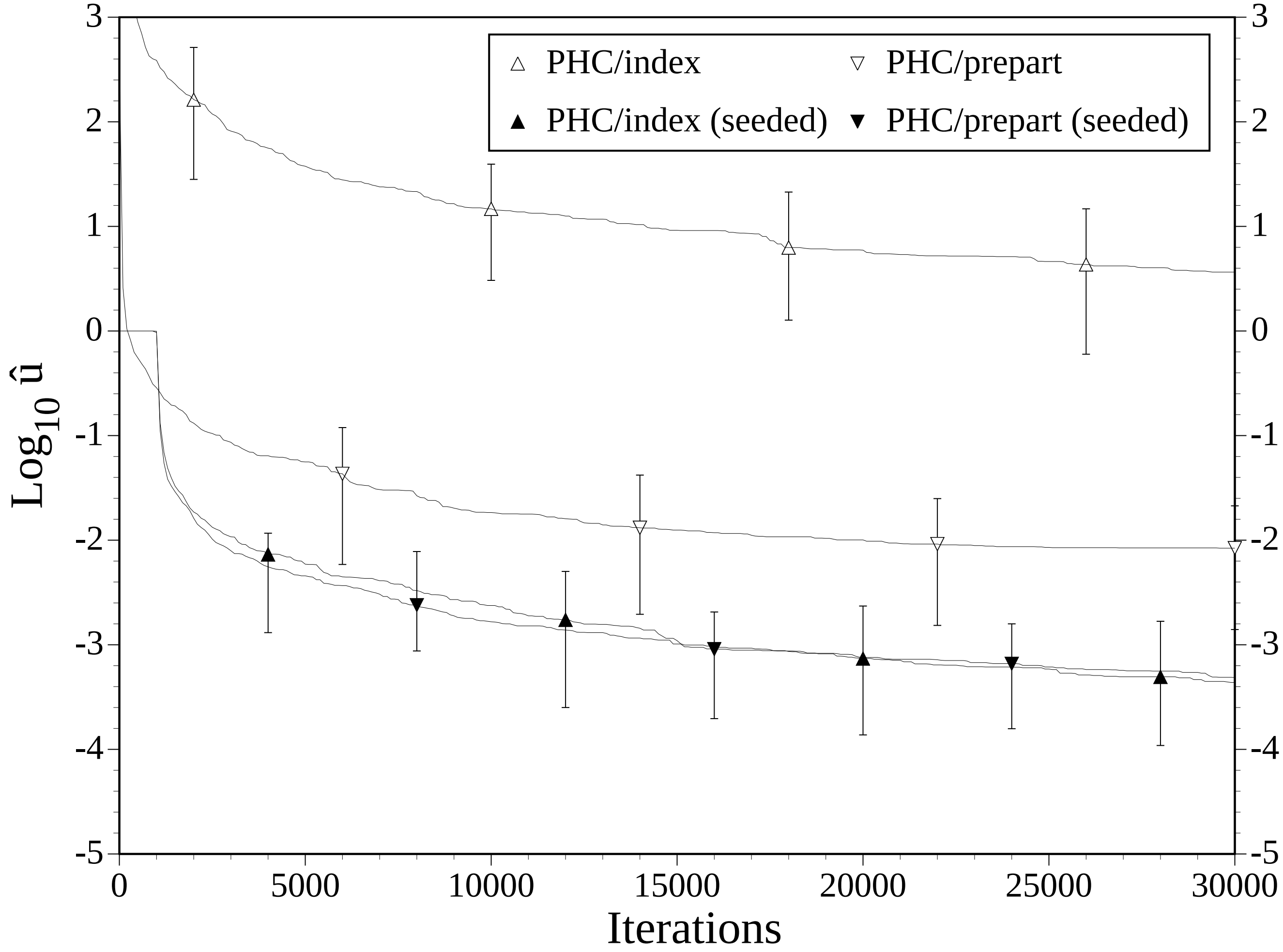


Performance for Other Search Engines



- Choice of representation is *much* more important than choice of search engine.

Effect of Seeding



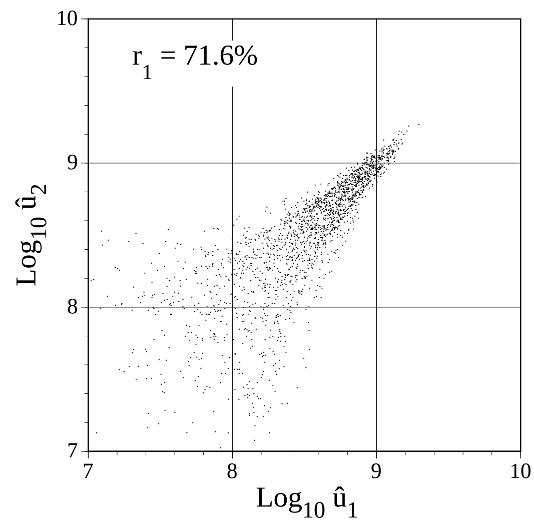
Empirical Analysis

- Three hypotheses to explain the effective performance of local search:

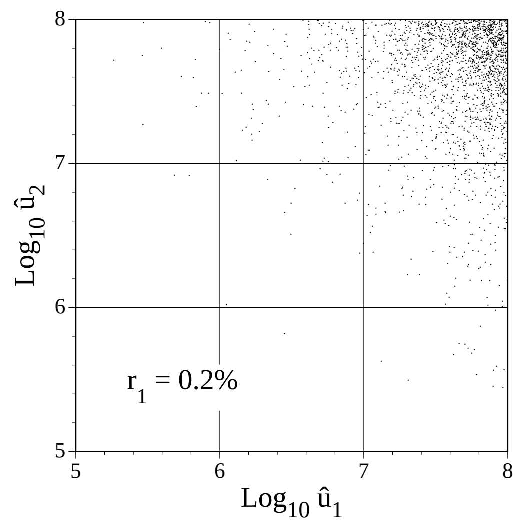
  1. Neighborhood structure
  2. A priori distribution of solutions
  3. "Gravitation" about KK solution

Neighborhood Structure

- Neighborhood structure exists in the direct encoding...

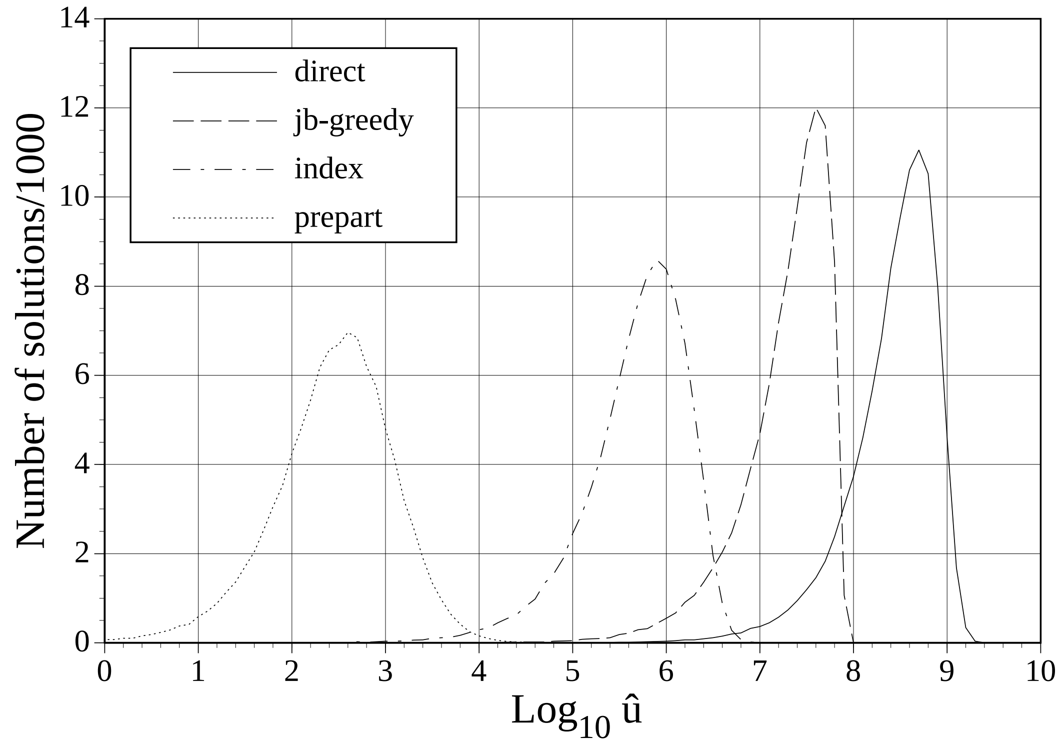


- but not where it counts:



- Same result for other encodings

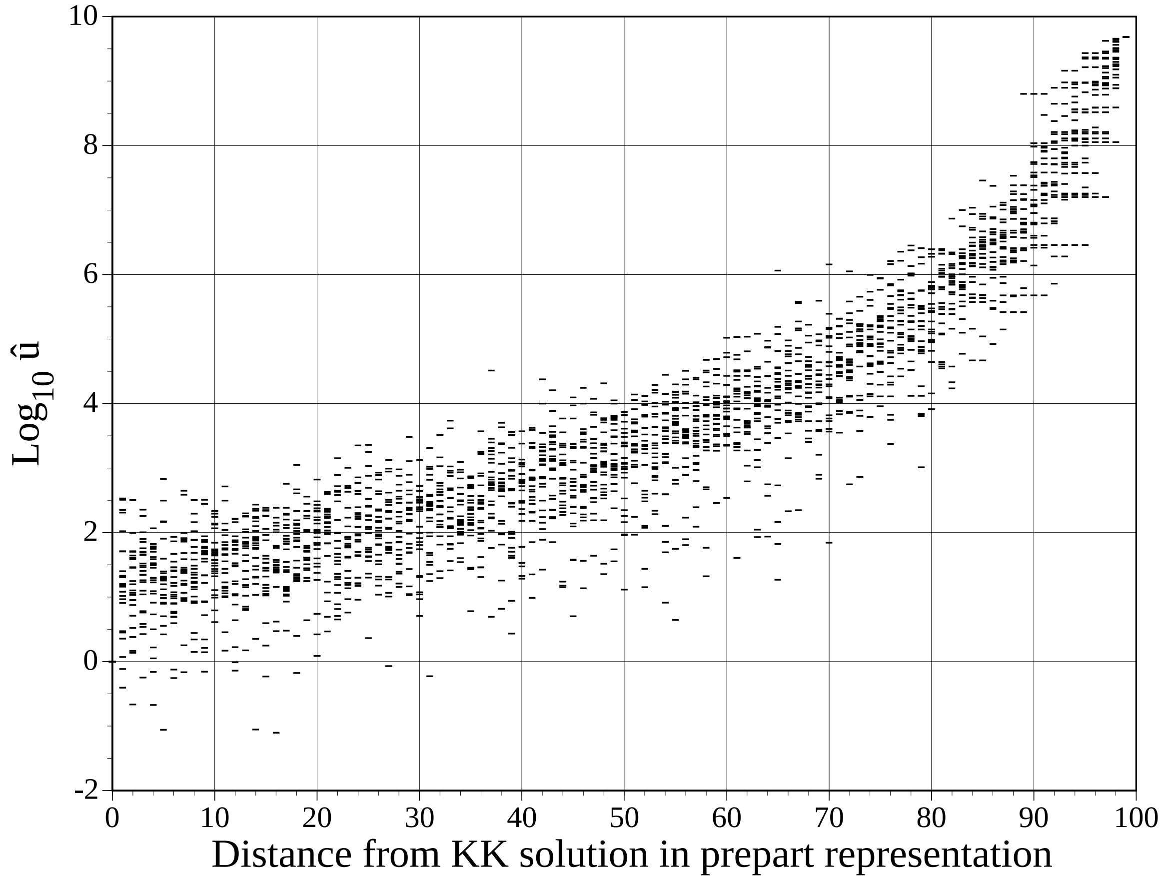
A Priori Distribution



- Big differences among the encodings, but doesn't predict
  - The superiority of search over RGT
  - The benefits of seeding

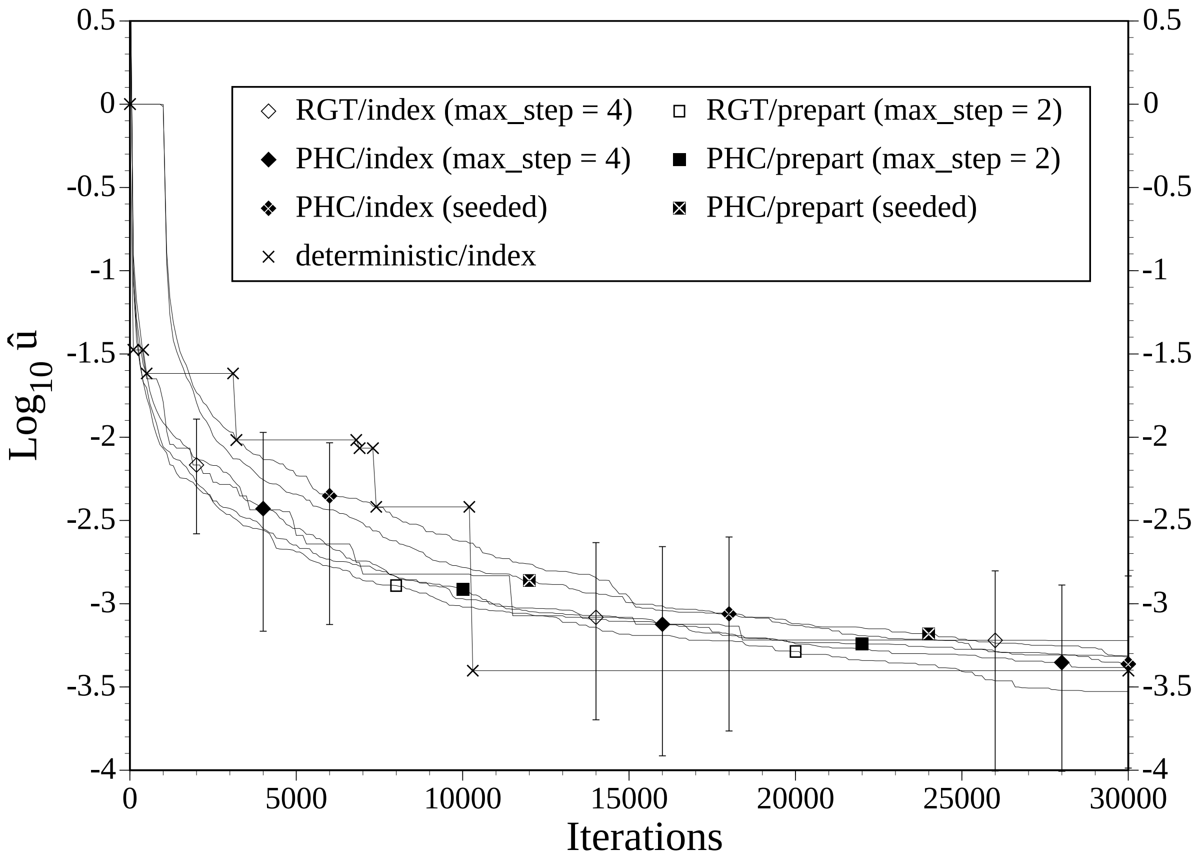
Gravitation

- A different kind of neighborhood structure:



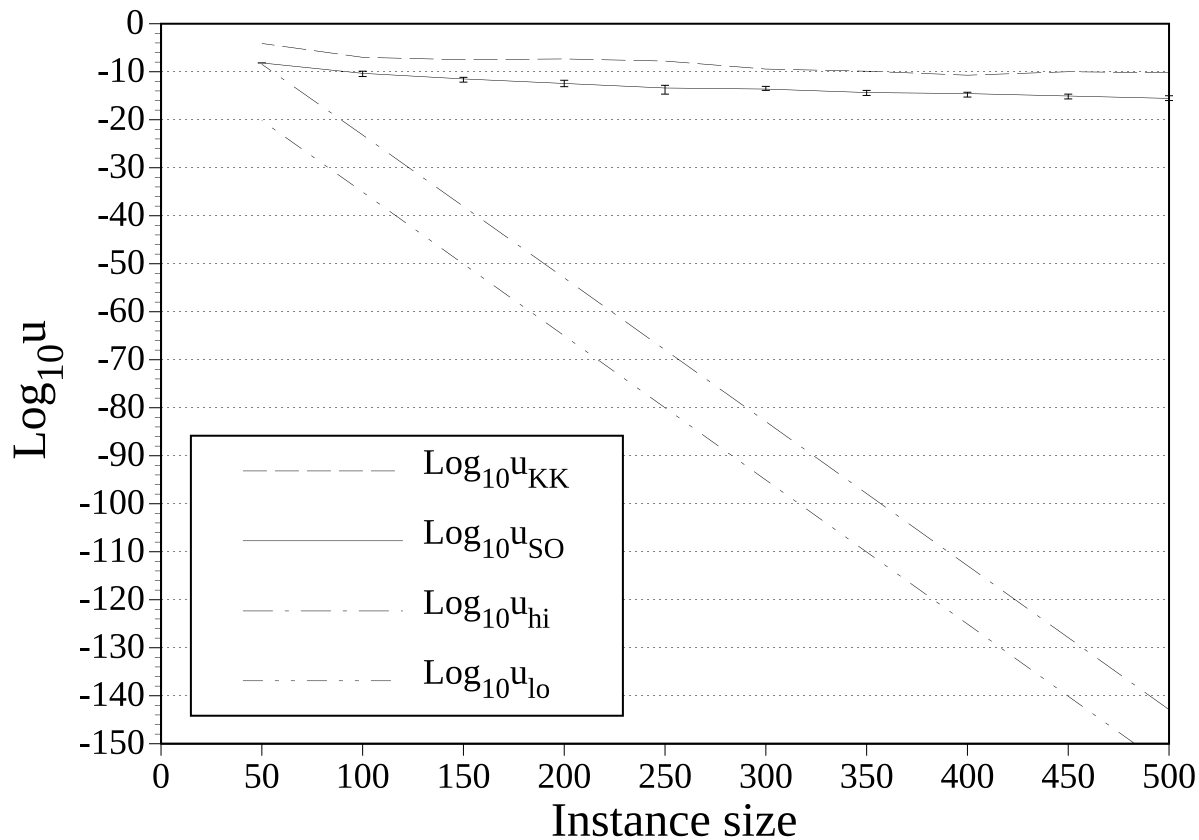
Additional Improvements

- Use gravitation to our advantage:
  restrict search space to the K neighborhood



Remaining Opportunities

- Performance of RGT/index (max_step = 4) vs. probable optimum:



The General Recipe

Given an NP-hard optimization problem:

- Find a good deterministic constructive heuristic solution method.
- Figure out how to parameterize the construction using:
  - Parameterized arbitration
  - Parameterized constraint
  - Parameterized greediness
- Apply stochastic search techniques to the parameterized heuristic.
- Take an empirical approach to refining and analyzing the resulting algorithms.