# A Fine-tuned Learning Approach to Optimization

by

Steven Coy, University of Maryland
Bruce Golden, University of Maryland
George Runger, Arizona State University
Edward Wasil, American University

Presented at INFORMS Atlanta, November 1996

# Fine-tuned Learning Approach

0. Start with original inputs.

1. Average, aggregate, approximate or smooth the inputs.

2. Perform PROCEDURE.

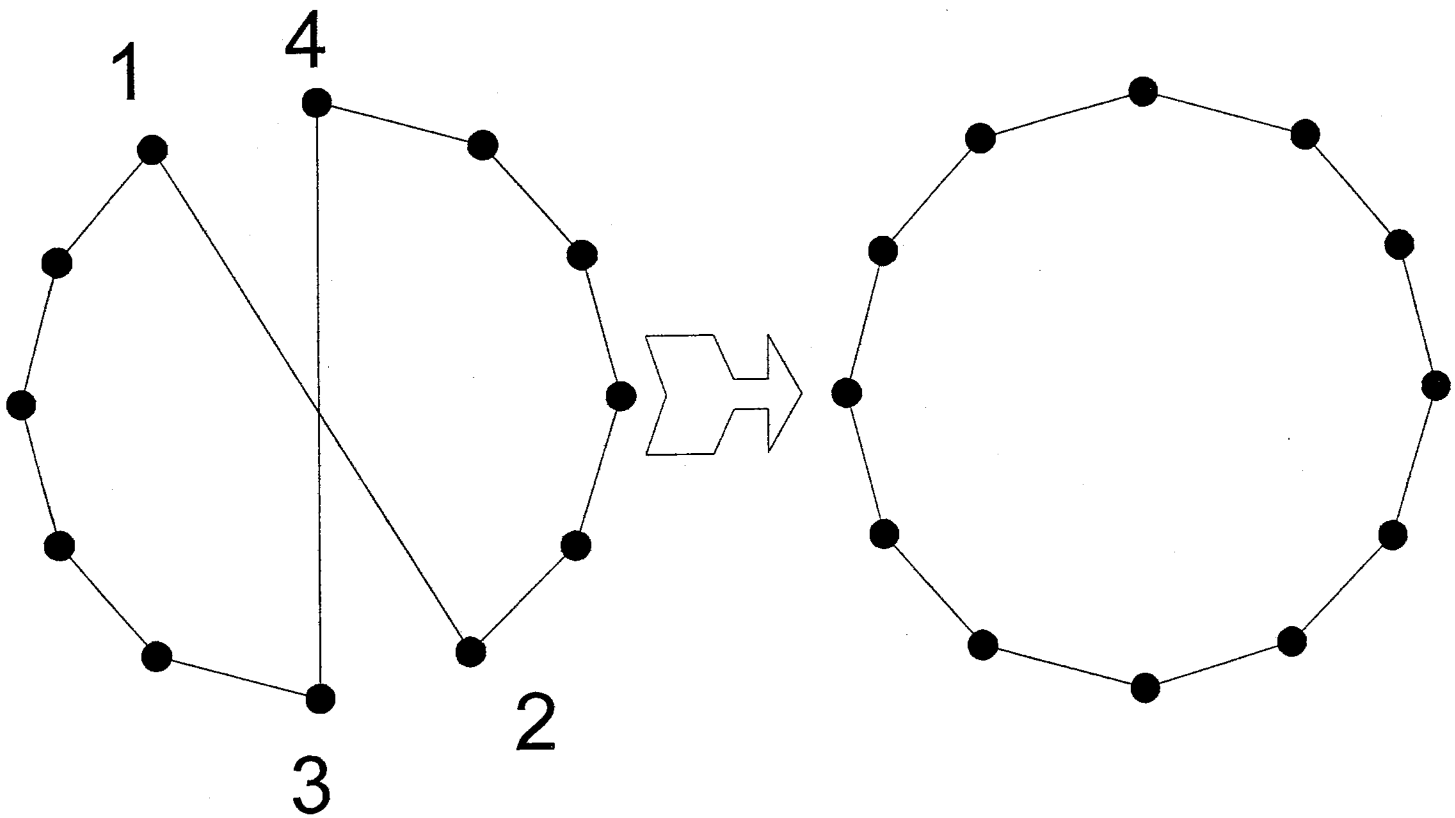
3. If inputs are the original inputs, then stop. Otherwise, disaggregate or refine the inputs.

4. Go to step 2.

# Applications of Fine-tuned Learning

- Prediction and classification
  - backpropagation, feed-forward neural networks
- Optimization
  - elastic nets
  - smoothing
  - noising

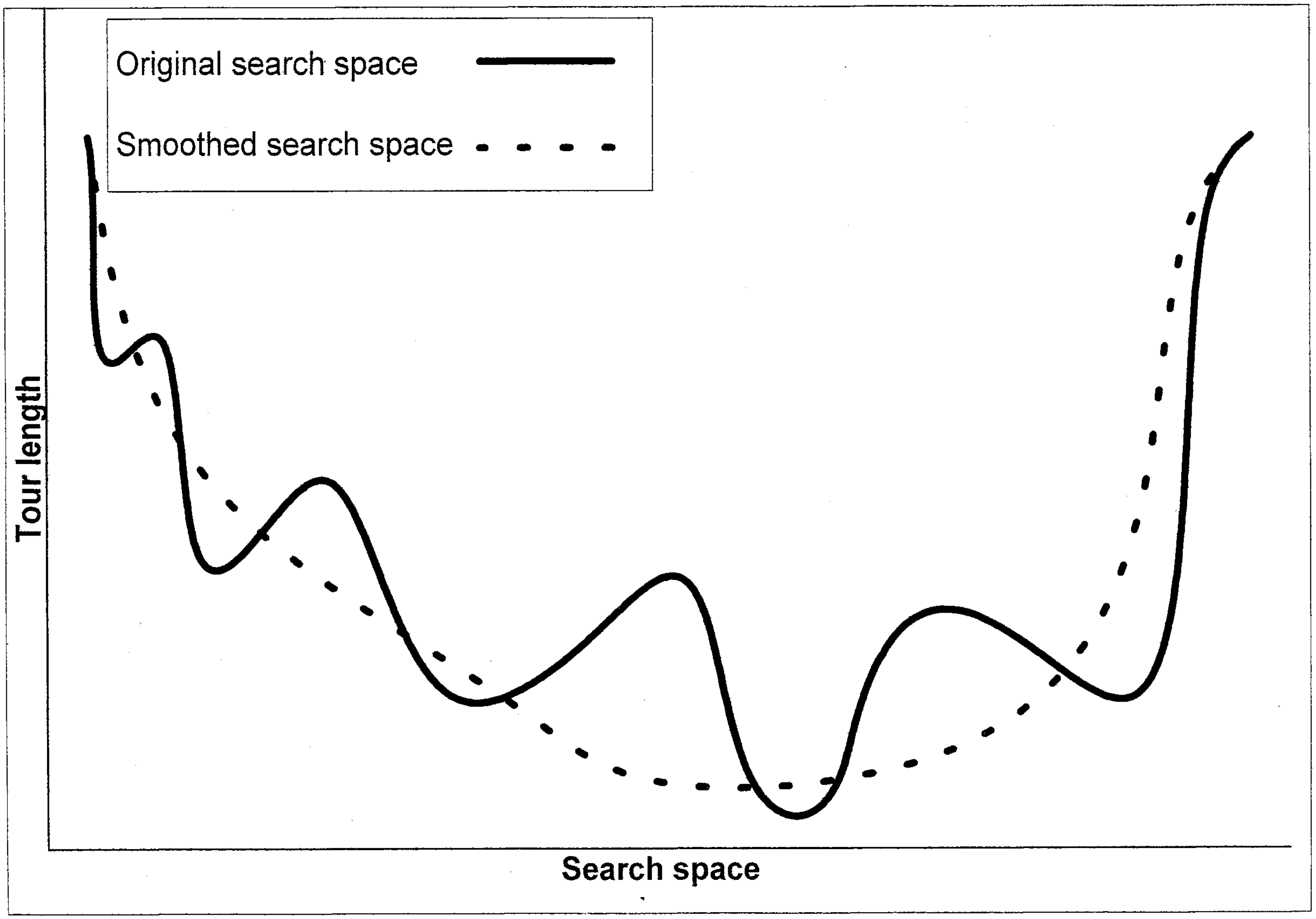
# Two-opt Exchange



# Overview

- Smoothing and noising are examples of fine-tuned learning.
- In this talk, we focus on smoothing.
- In smoothing, we first normalize distances so that they fall between 0 and 1.
- At each iteration, the original distances are transformed by the smoothing transformation. Two-opt is applied. The degree of transformation decreases from one iteration to the next, until the original distances re-emerge.

# Motivation for Search Space Smoothing



# Search Space Smoothing

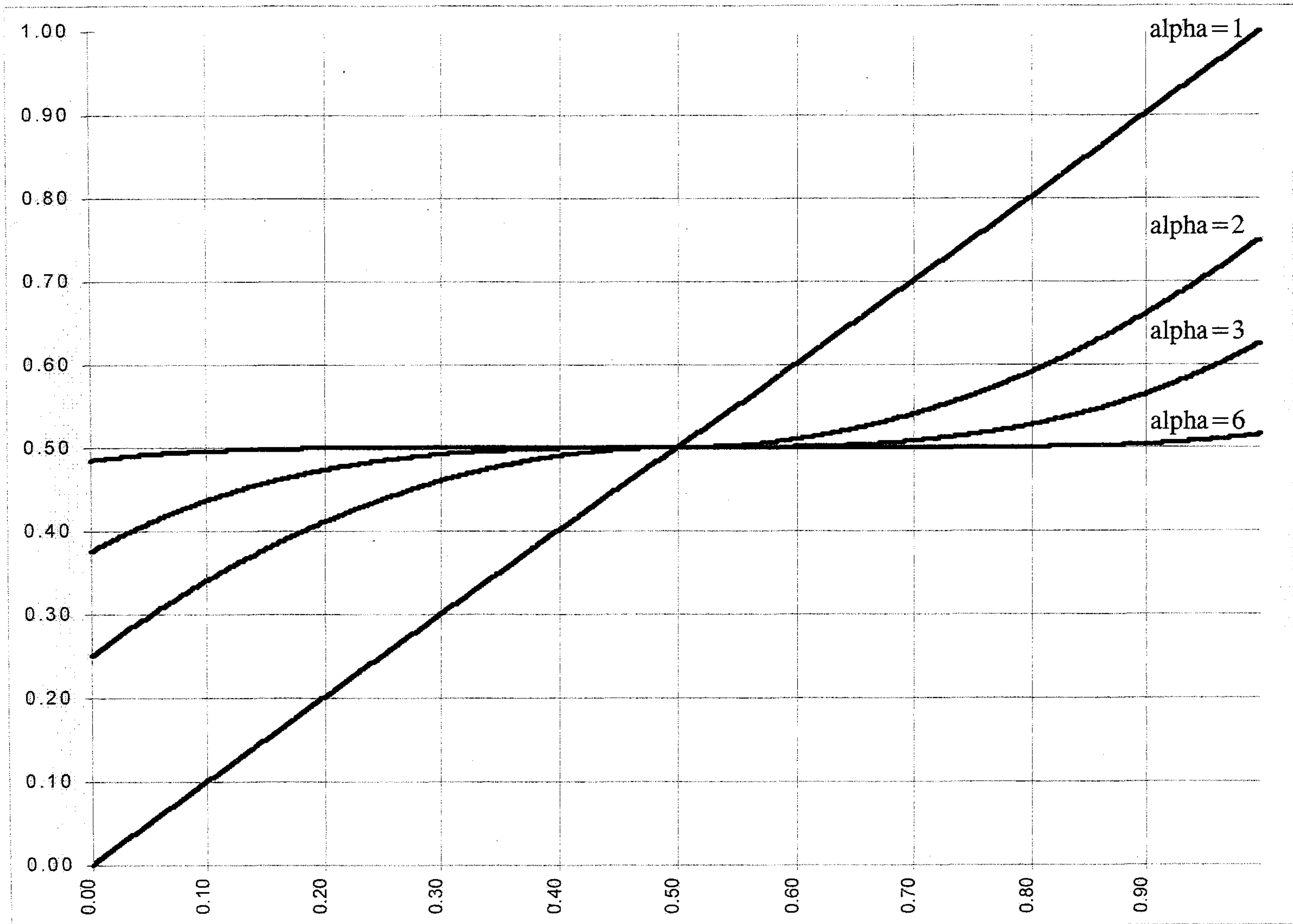
**[Gu and Huang 94]**

Transform:

$$d_{ij}(\alpha) = \begin{cases} \bar{d} + (d_{ij} - \bar{d})^{\alpha}, & d_{ij} \geq \bar{d} \\ \bar{d} - (\bar{d} - d_{ij})^{\alpha}, & d_{ij} < \bar{d} \end{cases}$$

- Iterations: $\alpha = 6, 3, 2, 1$
- The transform clusters distances in the center.
- The ranking of the distances is preserved.

# Smoothing Transformation



# 100 - node Instances

**Euclidean Instances**

| Algorithm | Tour length | Percent over HK | Run time (s) |
|---|---|---|---|
| Random Start | | | |
| Smoothing | 8.0037 | 3.91 | 0.152 |
| Two-opt | 8.2712 | 7.38 | 0.111 |
| Preprocessing time | | | 0.054 |
| Tour Construction | | | |
| Smoothing | 7.9616 | 3.36 | 0.112 |
| Two-opt | 8.0468 | 4.47 | 0.039 |
| Preprocessing time | | | 0.063 |

**Random Instances**

| Algorithm | Tour length | Percent over HK | Run time (s) |
|---|---|---|---|
| Random Start | | | |
| Smoothing | 2.5983 | 27.75 | 0.108 |
| Two-opt | 3.3055 | 62.52 | 0.004 |
| Preprocessing time | | | 0.051 |
| Tour Construction | | | |
| Smoothing | 2.5760 | 26.65 | 0.093 |
| Two-opt | 2.7248 | 33.97 | 0.013 |
| Preprocessing time | | | 0.049 |

# 1000 - node Instances

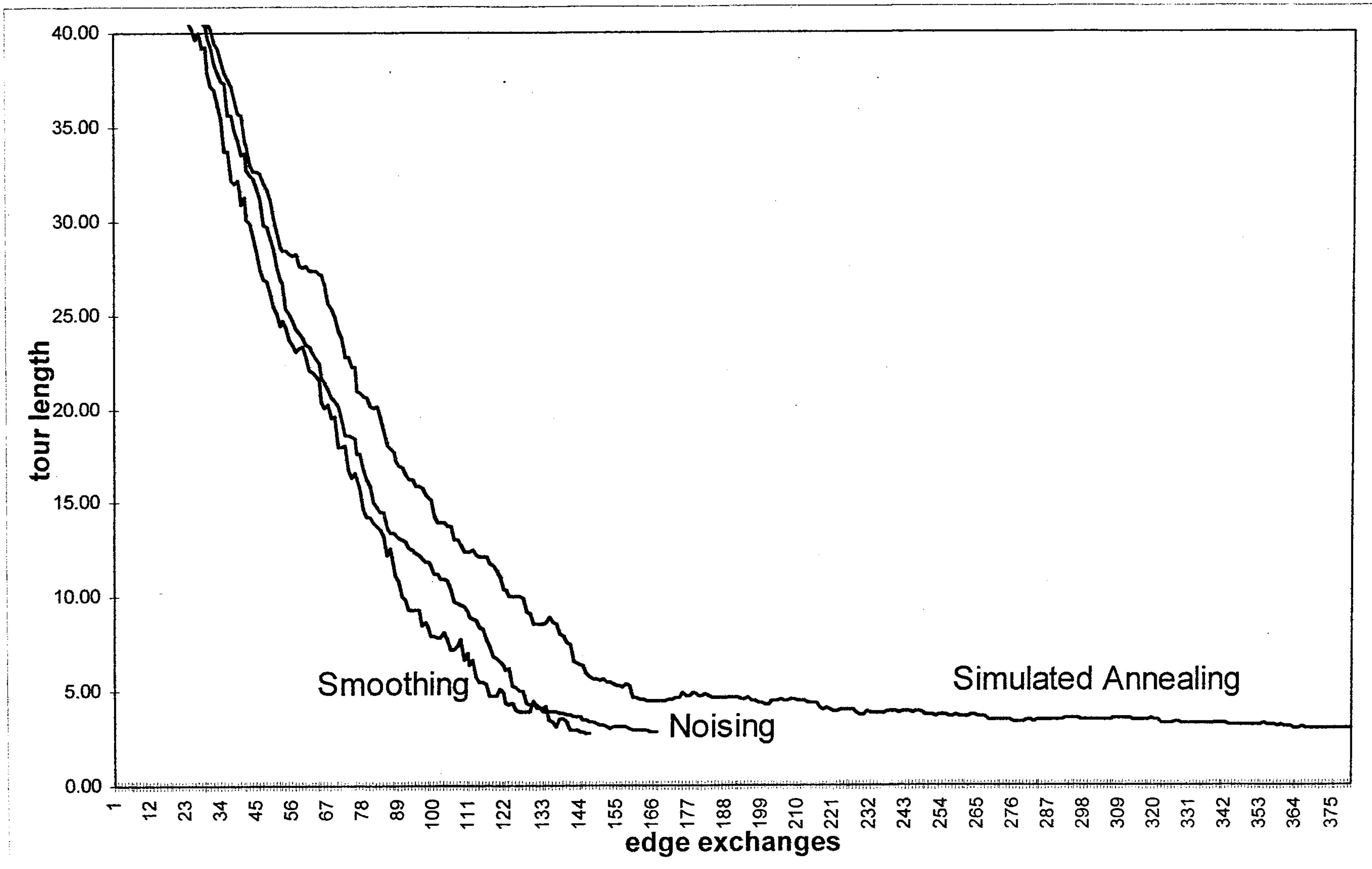
**Euclidean Instances**

| Algorithm | Tour length | Percent over HK | Run time (s) |
|---|---|---|---|
| Random Start | | | |
| Smoothing | 24.4830 | 6.67 | 12.201 |
| Two-opt | 25.3392 | 10.40 | 8.883 |
| Preprocessing time | | | 9.230 |
| Tour Construction | | | |
| Smoothing | 24.1652 | 5.29 | 9.709 |
| Two-opt | 24.3102 | 5.92 | 2.750 |
| Preprocessing time | | | 8.800 |

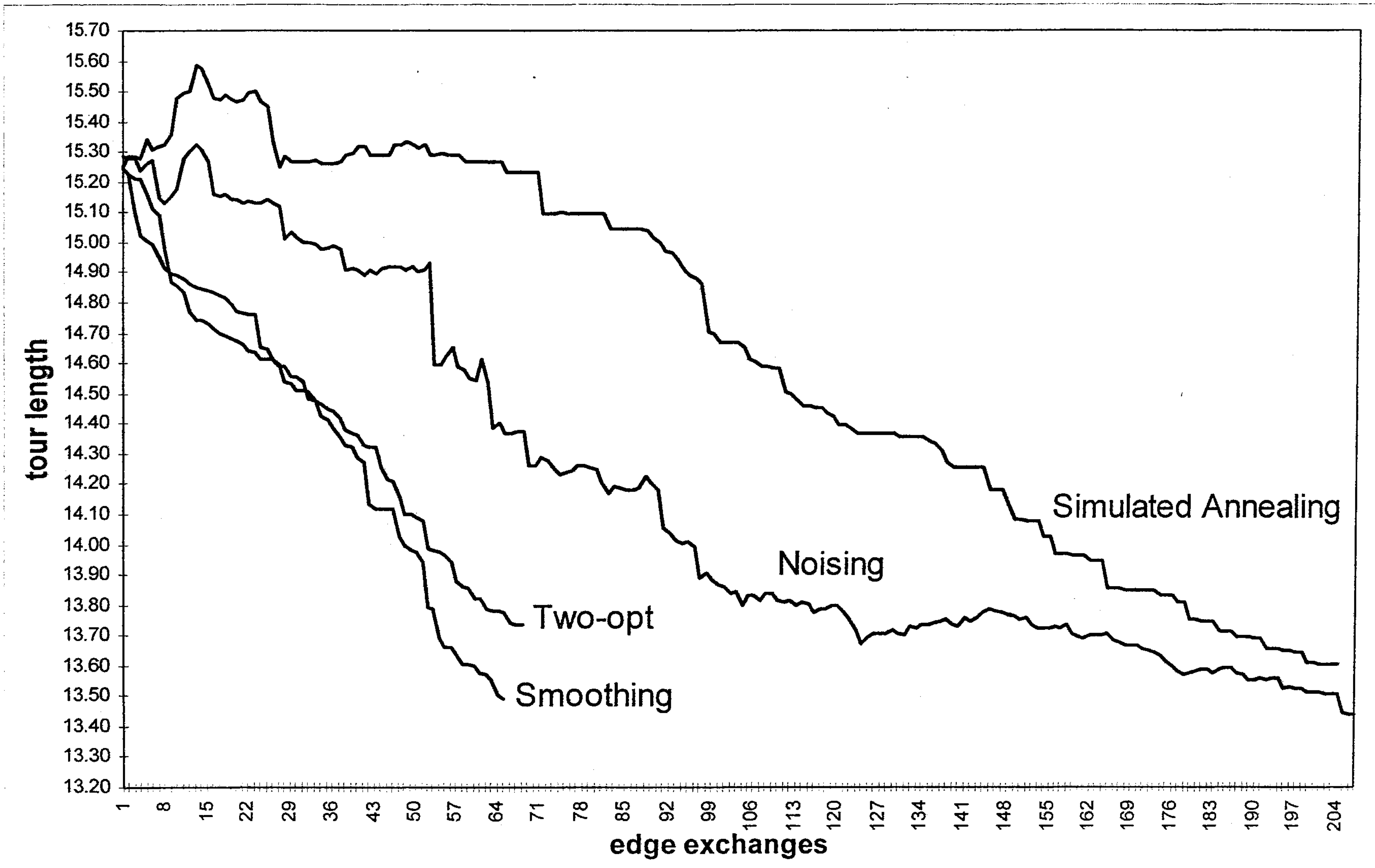
**Random Instances**

| Algorithm | Tour length | Percent over HK | Run time (s) |
|---|---|---|---|
| Random Start | | | |
| Smoothing | 4.1559 | 103.56 | 8.188 |
| Two-opt | 6.0135 | 194.55 | 1.724 |
| Preprocessing time | | | 9.000 |
| Tour Construction | | | |
| Smoothing | 3.3628 | 64.71 | 7.639 |
| Two-opt | 3.4291 | 67.96 | 0.616 |
| Preprocessing time | | | 9.000 |

# 100 - node Random Instance with a Random Start



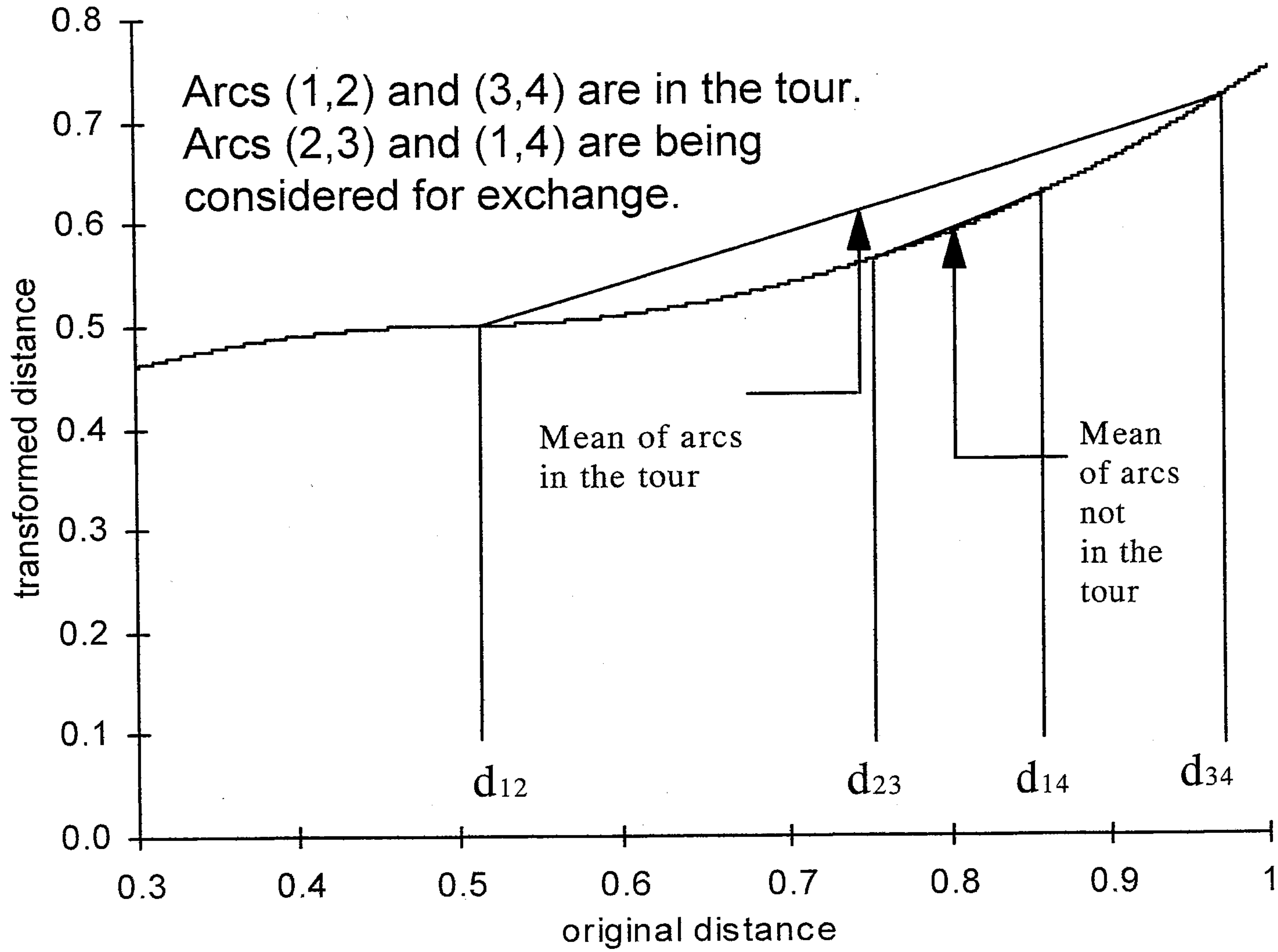
# 316 - node Euclidean Instance with a Greedy Start



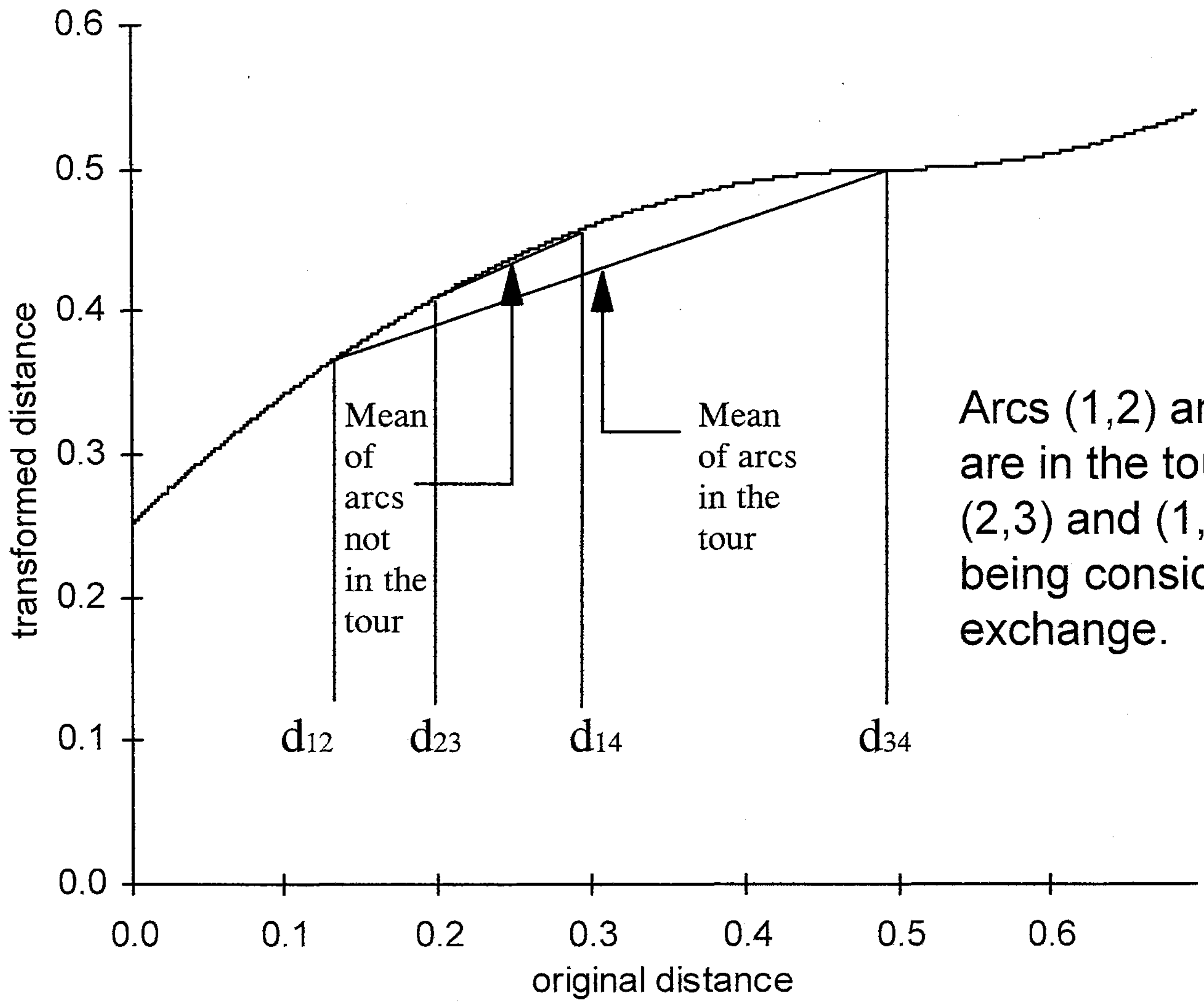
# Why Smoothing Works

- The transition from one transformed distance matrix to the next opens up new opportunities for exchanges.
- Smoothing allows some uphill moves.
- Smoothing allows the rejection of some marginally improving moves in favor of better moves.

# Unfavorable Moves in the Convex Region of the Transformation

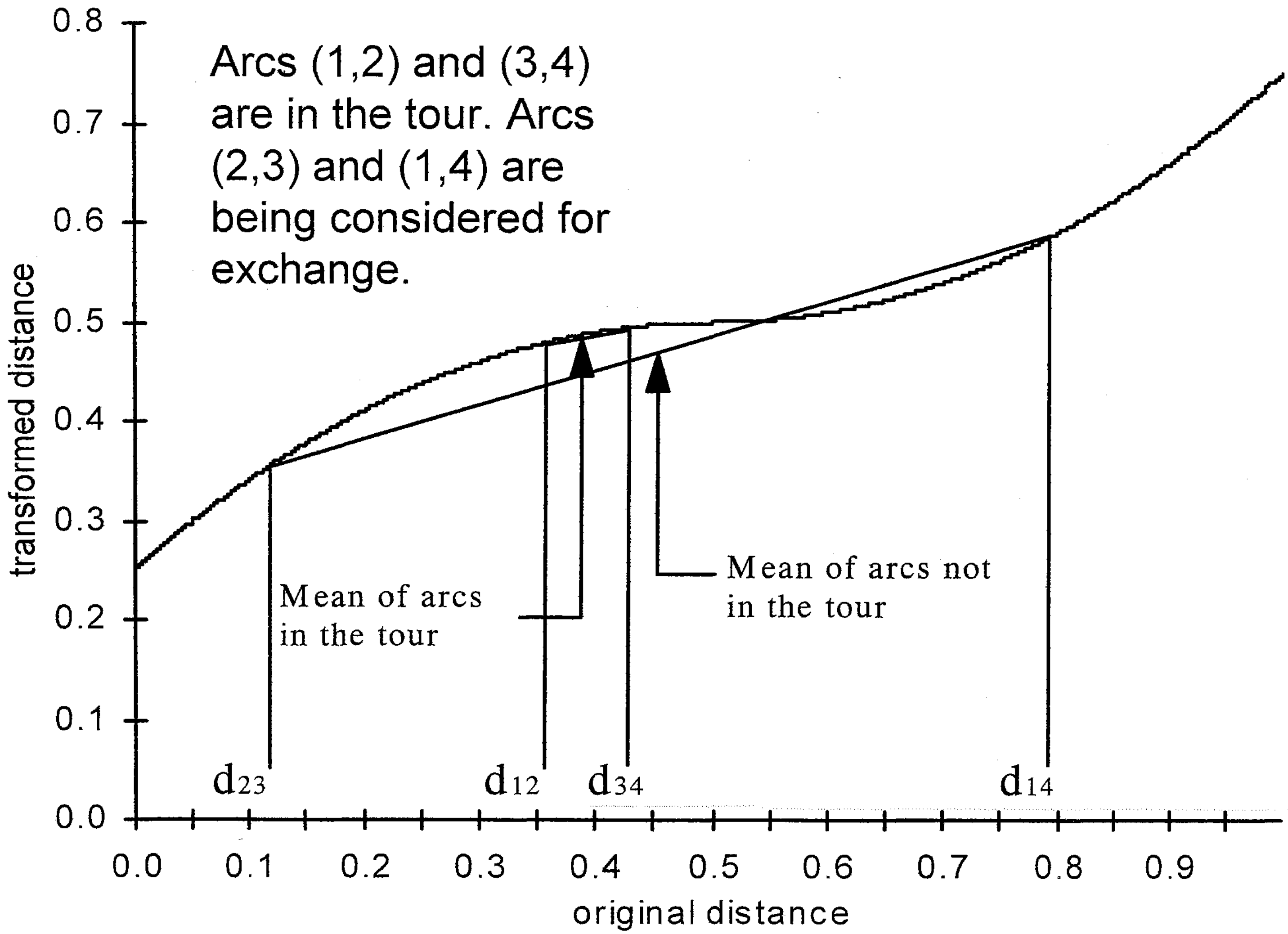


# Marginal Improvements Rejected in the Concave Region of the Transformation




Arcs (1,2) and (3,4) are in the tour. Arcs (2,3) and (1,4) are being considered for exchange.

# Unfavorable Moves in the Transitional Region of the Transformation



# Comparison of Two-opt Implementations

| | | Downhill Moves | |
|---|---|---|---|
| | | Always Accept | Occasionally Reject |
| Uphill Moves | Never Accept | Two-opt<br>Three-opt<br>Or-opt | Restricted smoothing |
| | Occasionally Accept | Simulated annealing<br>Record-to-record travel<br>Great deluge<br>Threshold accepting | Smoothing<br>Noising<br>Tabu search |

# Conclusions and Future Work

- We have introduced the notion of FTL and applied it to examples from prediction and optimization. We are currently applying FTL to classification problems as well.
- We have used FTL as a mechanism for connecting neural networks and combinatorial optimization.
- We have developed a new way of classifying heuristics.
- Smoothing clearly outperforms two-opt and takes approximately the same amount of time.
- Further experimentation with different “smoothing” functions has lead to even better results.