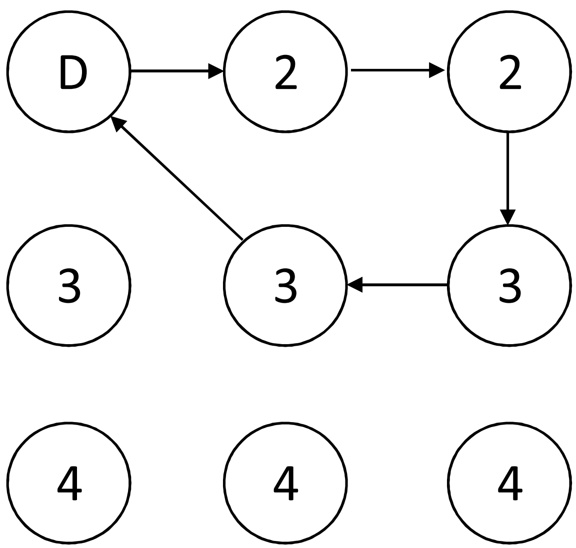
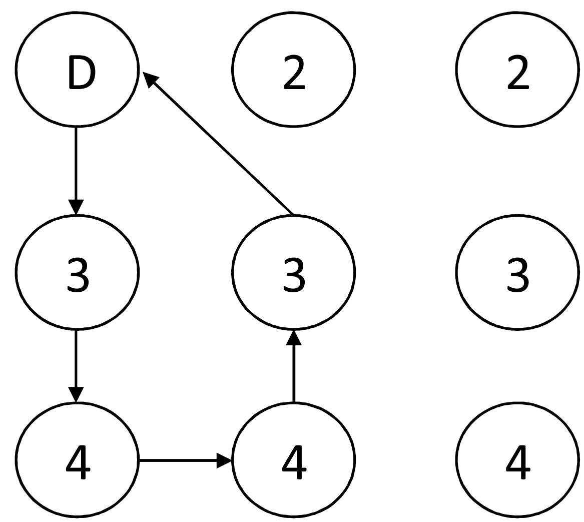
The Effective Application of a New Approach to the Generalized Orienteering Problem

John Silberholz
Bruce Golden
R.H. Smith School of Business
University of Maryland, College Park

The Orienteering Problem

- Score associated with each node
- Goal: Maximize score collected in cycle subject to distance limit





- Score: 10

Score: 14

The Generalized Orienteering Problem

- Each node has multiple attribute scores
- A function of the scores determines an overall score for a cycle
- *m* attributes, each with weight W_m
  - Weights sum to 1
- Node j has score $S_i(j)$ for attribute *i*
- Benchmark problems involve touring
- Score for set of nodes, N, in a cycle:

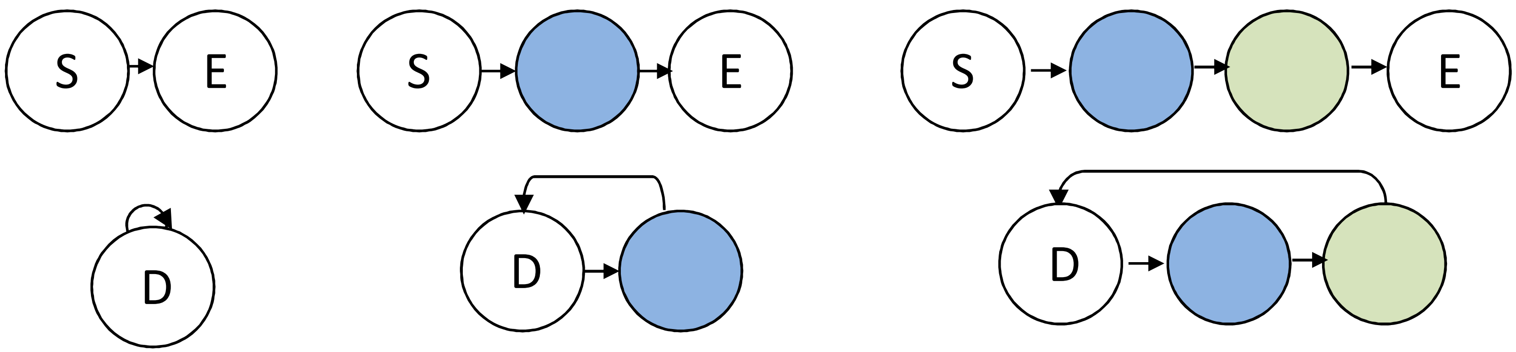
$$S_N = \sum_{i=1}^{m} W_i \left[\sum_{j \epsilon N} \{S_i(j)\}^k\right]^{1/k}$$

The 2-Parameter Iterative Algorithm

- Developed with simplicity in mind
- Based on a Process *P*
  - Maintains a single solution
  - Iteratively perturbs and then improves the solution
- Process *P* is run until it returns a result worse than a previous run

Generate an initial solution

- Begin with start and end nodes
- Add new nodes directly before the end node, choose from 4 random selections
- Minimize total length when adding

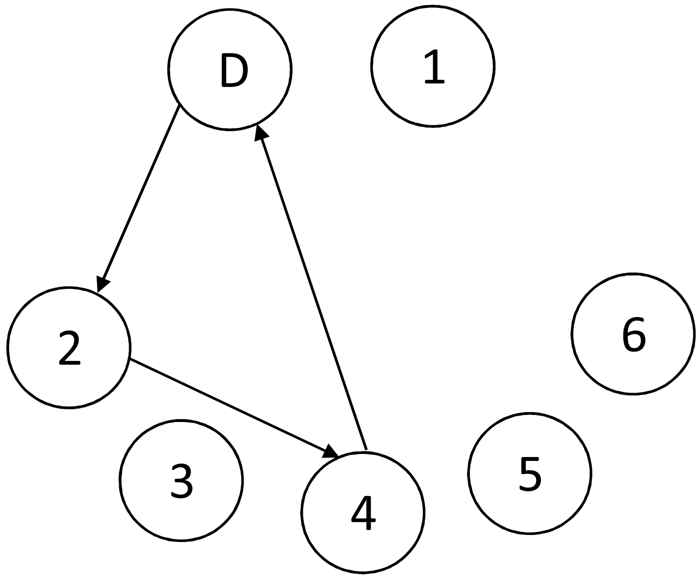


- Perform all available 2-opts
- Perform path tightening

Path tightening

- Order unused nodes based on how much adding them would improve overall score
- Iterate this list, adding when possible
  - Insert in a best-fit manner
- Designed to be fast
  - Better results for GOP if list is recreated each addition (but not for OP)
  - That approach would take longer

Path tightening example

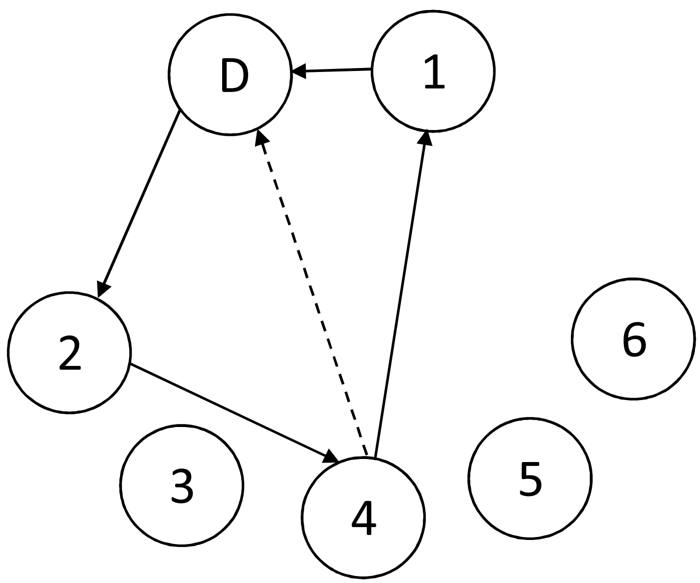


6

1

3

5



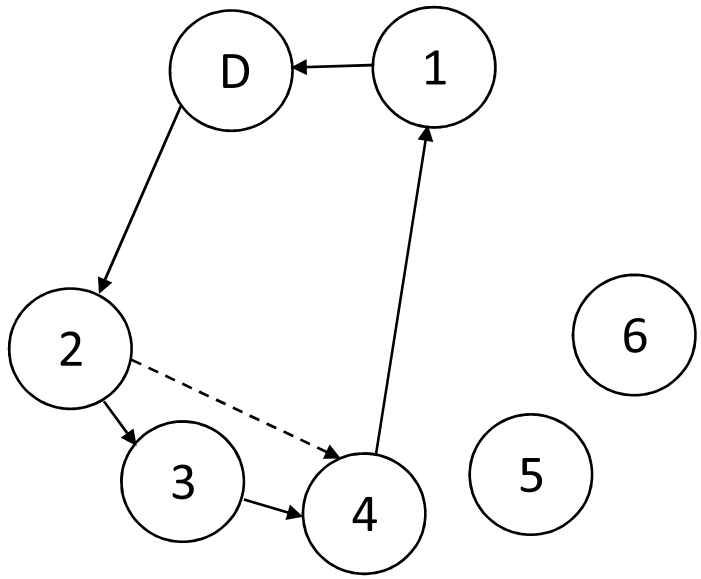
6 « not added

1 « added

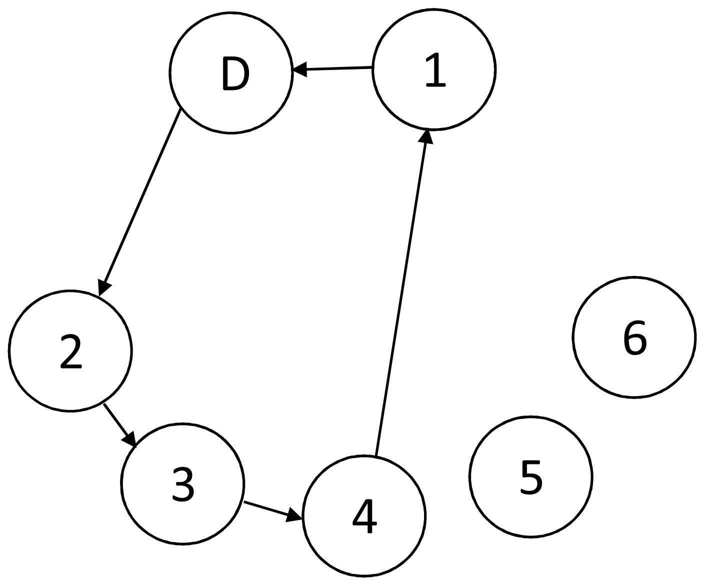
3

5

Path tightening example



6
1
3
5



6
1
3
5

Perturbation and improvement

- Perturb solution
  - Remove 4 random nodes
- Improve solution
  - Path tightening with 4 removed nodes on bottom of add list (for diversity)
  - All available 2-opts
  - Unrestricted path tightening
  - All available 2-opts
- Repeat until 4500 runs without improvement

Performance on GOP instances

k	WGW-GA	ANN	HS
1	0/0	0/0	--
3	2/0	2/0	--
4	4/0	2/0	--
5	4/0	3/0	2/0
10	0/0	4/0	--
Total	10/0	11/0	2/0

- Outperformed others in solution quality
  - Never outperformed on an instance
- Ran quickly (< 1 sec.)
  - Faster than other algorithms
- Need for larger test instances

Performance on small OP instances

n	Number of inst.	TA	CR
32	18	11/0	0/0
21	11	7/0	0/0
33	20	20/0	0/0
66	26	13/1	7/1
64	14	13/1	4/1
Total	89	64/2	11/2

- Outperformed others in solution quality
- Ran quickly (<1 second)
  - Same normalized speed as TA
  - Slower than CR, but better runtime growth
- Instances still too small

Performance on TSPLib-based OP instances

Size range	Number	TS Err.	TS Sec.	2-P IA Err.	2-P IA Sec.
≤90	24	0.45%	1.36	0.19%	0.72
91-130	42	2.14%	2.99	1.71%	2.44
131-200	33	5.13%	5.68	3.61%	6.01
201-400	27	9.94%	19.53	6.62%	21.28

- Optimal solutions from Fischetti et al. paper
  - 116 optimal, 10 timed out after 5 hours
- 2-P IA outperforms best OP heuristic to date for solution quality
- Runtimes are comparable
  - Run on same hardware

Variability to seed

Size Range	Min-of-5 Error	Max-of-5 Error
≤90	0.14%	0.66%
91-130	0.49%	3.01%
131-200	2.65%	5.65%
201-400	3.61%	7.96%

- Reasonably large variability to seed
  - Caused by greedy mechanisms in 2-P IA
- Disadvantage if algorithm is run only once
  - One instance had min-of-5 error of 1.1%, max-of-5 error of 30.2%
- Can help if multiple runs per instance
  - Found a new best solution for one instance

Conclusions

- 2-P IA is simple
- 2-P IA outperforms all OP and GOP heuristics for solution qualities
- 2-P IA is fast, competitive in runtime with all other algorithms considered
- GOP test base must be improved
    - Current largest problem has only 27 nodes
    - Only one non-linear function being tested