Plowing with Precedence

A Variant of the Windy Postman Problem

Benjamin Dussault, Bruce Golden, Edward Wasil, and Chris Groer

Overview

- The Chinese Postman Problem (CPP)
- The Windy Postman Problem (WPP)
- Plowing with Precedence
    - Literature Review
    - Introduction
    - Problem Statement
    - Problem Formulation
    - Solution Methodology
    - Results
    - Conclusions

The Chinese Postman Problem (CPP)

- Consider a graph $G=\{V,A\}$ where
  - $V=\{v_i\}$
  - $A=\{(v_i,v_j) \mid v_i,\ v_j \in V\}$
  - c_{ij} = Cost of traversing on arc (v_i,v_j)
  - $c_{ij} = c_{ji}$
- Goal: To construct a least-cost tour which visits all arcs in A at least once

The Windy Postman Problem (WPP)

- A close variant of the Chinese Postman Problem
- The graph is “Windy”; it is harder to traverse in one direction on an arc as opposed to the other
- Goal: To construct a least-cost tour which visits all arcs in A at least once
- Key Difference: Costs are not symmetric

Solution Methodology of CPP and WPP

- Crucial observation: If graph is Eulerian, then an optimal tour can readily be obtained using Fleury's Algorithm
- It is therefore sufficient to convert the instance graph to a Eulerian graph in an optimal way
- Possible methods
  - Integer Programming
  - Add least-cost paths between odd-degree nodes

Plowing with Precedence

Literature Review

- Arc Routing is well studied. There are many summaries:
  - Eiselt et al. (1995a, 1995b)
  - Assad and Golden (1995)
  - Dror (2000)
- Perrier et al. (2006, 2007) provide a four-part summary of winter road maintenance covering:
  - System Design
  - Models and Algorithms
  - Vehicle Routing and Depot Location
  - Vehicle Routing and Fleet Sizing

Plowing with Precedence

Introduction

- Variant of the Windy Postman Problem
- Non-symmetric costs correspond with the difficulty of plowing uphill
- Once a street is plowed, the cost of subsequent traversals is significantly less
  - Requires two new costs for each arc: the cost of deadhead in each direction
  - Introduces the concept of precedence: the cost of a street now depends on wether it has been traversed already

Plowing with Precedence

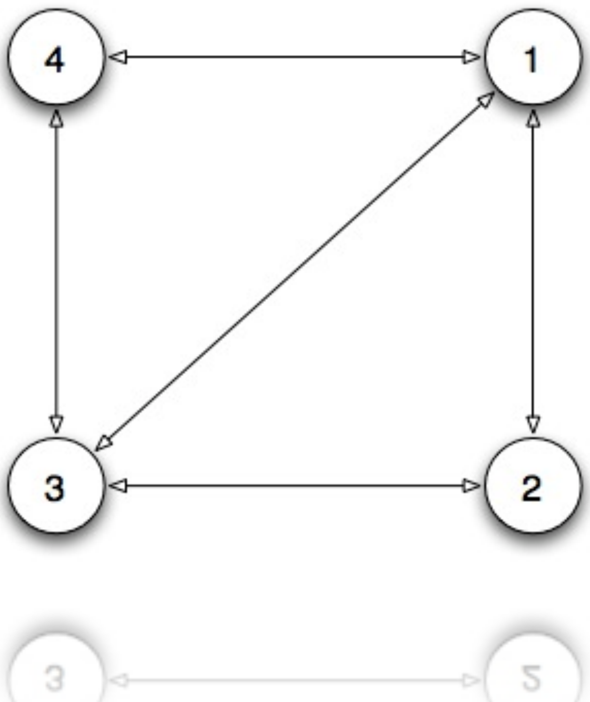
Introduction

- The concept of precedence requires a fundamentally different solution methodology than that used in previous WPP literature
- A Eulerian graph yields many Eulerian tours
  - Equivalent in WPP
  - Not equivalent in Plowing with Precedence

Plowing with Precedence

Introduction

Deadhead costs = 1

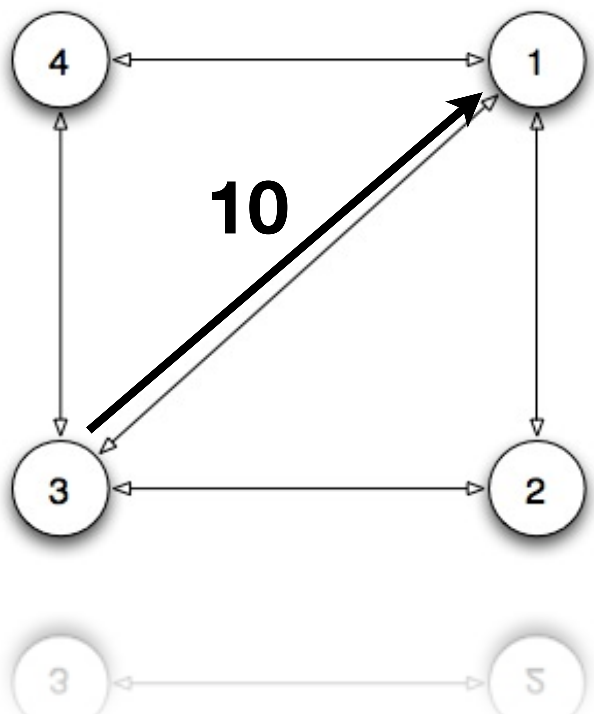


Original Instance

Plowing with Precedence

Introduction

Deadhead costs = 1

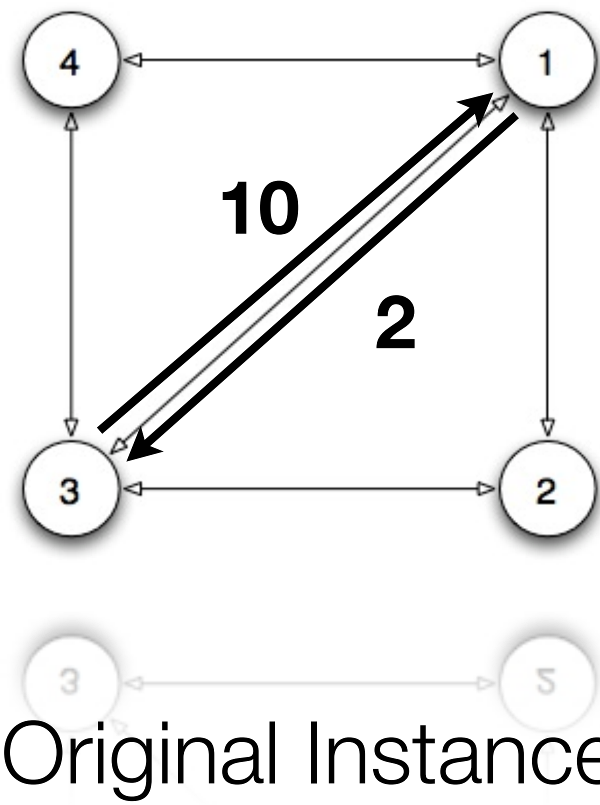


Original Instance

Plowing with Precedence

Introduction

Deadhead costs = 1

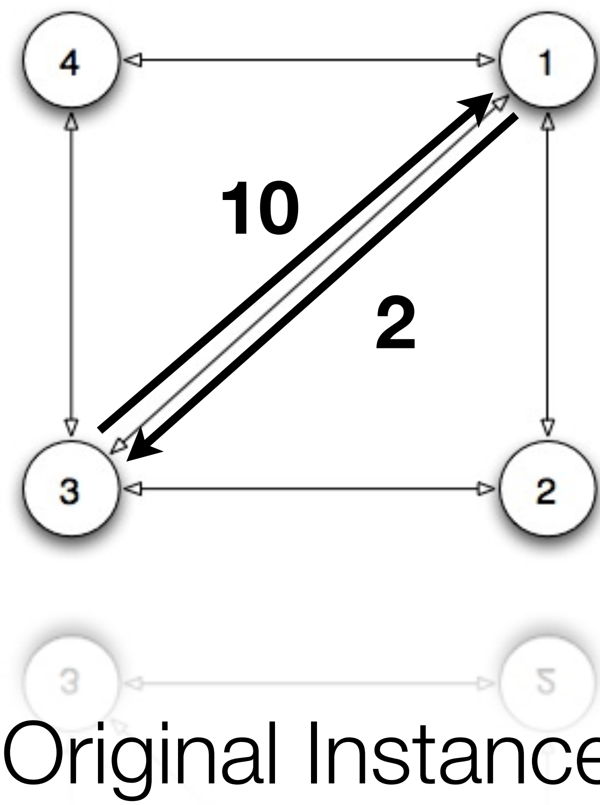


Original Instance

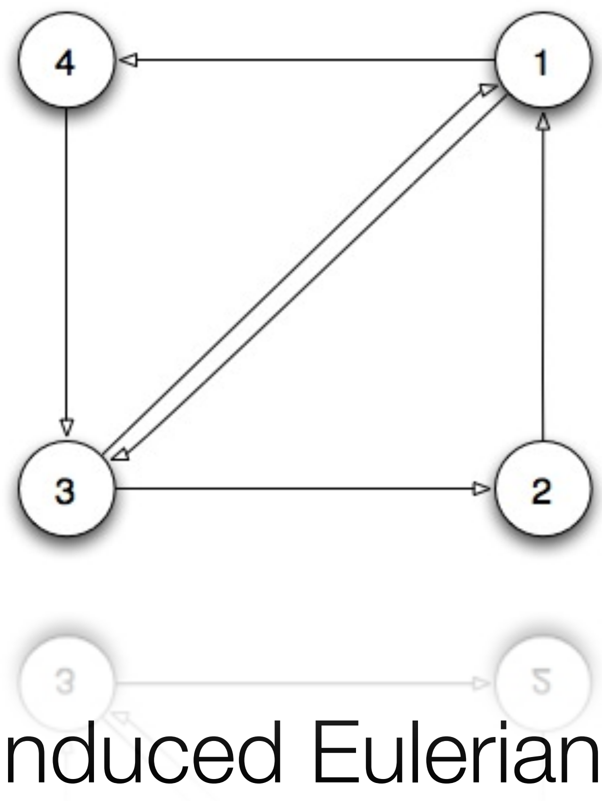
Plowing with Precedence

Introduction

Deadhead costs = 1



Original Instance

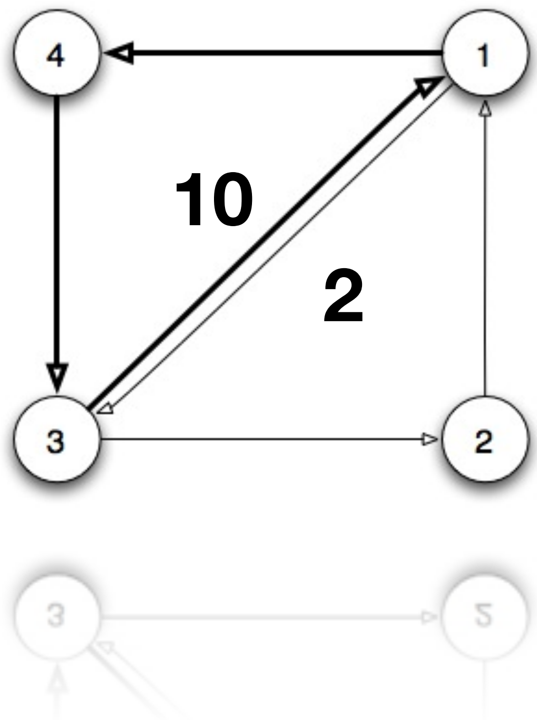


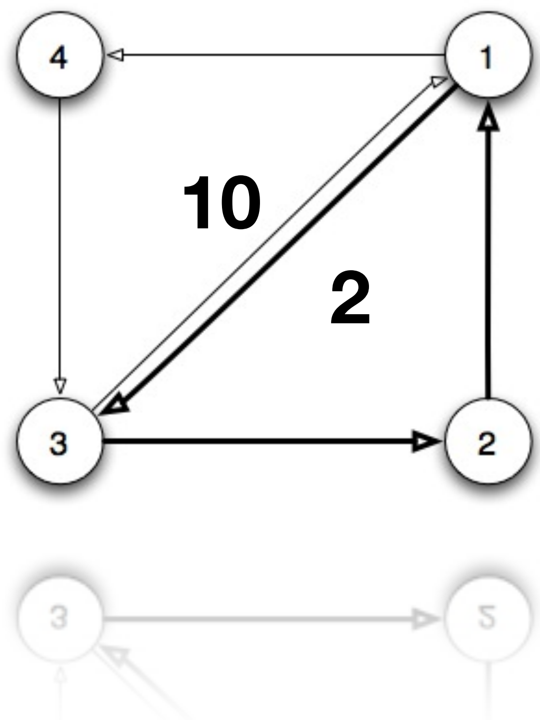
Induced Eulerian Graph

Plowing with Precedence

Introduction

- Multiple tours:
  - {1,4,3,1,3,2,1}
    - Travels arc (3,1) before (1,3)
  - {1,3,2,1,4,3,1}
    - Travels arc (1,3) before (3,1)





Plowing with Precedence

Problem Statement

- ❖ Consider a graph $G=\{V,A\}$ where
  - $V=\{v_i\}$
  - $A=\{(v_i,v_j) \mid v_i,\ v_j \in V\}$
  - c_{ij}^{+} = Cost of plowing on arc (v_i,v_j)
  - c_{ij}^{-} = Cost of deadheading on arc (v_i,v_j)
  - $c_{ij}^{+} >> c_{ji}^{+} >> c_{ij}^{-} \geq c_{ji}^{-}$
- ❖ Goal: To construct a least-cost tour which visits all arcs in A at least twice (once for each side of the street) and begins and ends at a depot (required to incorporate precedence)

Plowing with Precedence

Problem Statement

- Non-directed arcs allow plowing against the flow of traffic
- Good solutions will attempt to plow downhill both times
- Allows for the intriguing possibility of:
  - Plowing downhill
  - Then deadheading uphill
  - Then plowing downhill

Plowing with Precedence

Problem Formulation

- Requires an index *t* to incorporate precedence
- Essential elements:
  - x_{ijt} = 1 if plow (*i,j*) at time t, 0 otherwise
  - y_{ijt} =1 if deadhead (*i,j*) at time t, 0 otherwise
  - φ_{ijt} =1 if (*i,j*) is first plowed at time t, 0 otherwise
- Essential constraints:
  - Tour continuity
  - Forbid deadhead on (*i,j*) until (*i,j*) or (*j,i*) is plowed
- Large number of variables and constraints (~8000 and 19000 respectively for an instance with 10 arcs and 7 nodes)

Plowing with Precedence

Solution Methodology

- ❖ Construct solution framework using integer programming
  - Objective seeks to minimize framework tour cost
  - Solution serves as a lower bound
- ❖ Use solution framework to construct initial solution using Fleury's Algorithm
- ❖ Perform local search on obtained solution
  - Reinitialize and repeat local search
- ❖ Prune obtained solution to obtain final solution

Plowing with Precedence

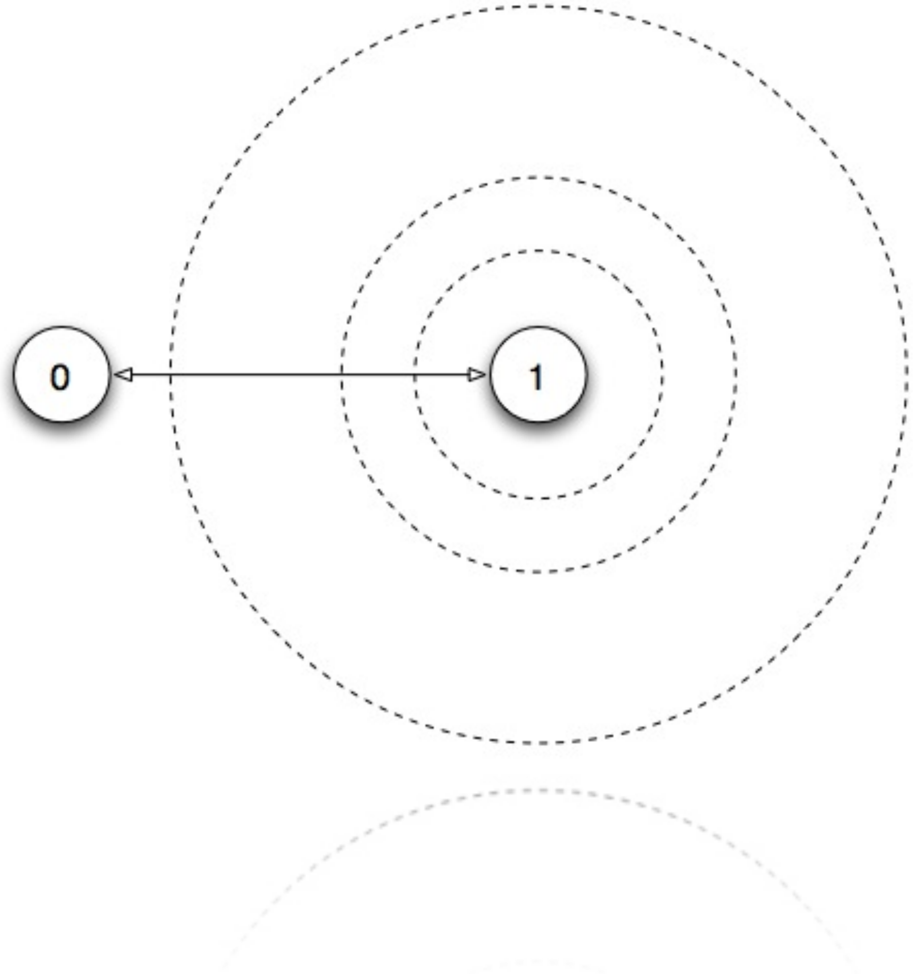
Solution Methodology - Solution Framework

- Adapt IP formulation for Windy Postman Problem
  - Ignores the concept of precedence, otherwise solves the problem
  - Objective function, which seeks to minimize ideal tour cost, serves as a useful lower bound
- Essential variables:
  - x_{ij} = the ideal number of times (i,j) is plowed
  - y_{ij} = the ideal number of times (i,j) is deadheaded
- Essential constraints:
  - Plow each street twice
  - Degree matching for each node

Plowing with Precedence

Solution Methodology - Solution Framework

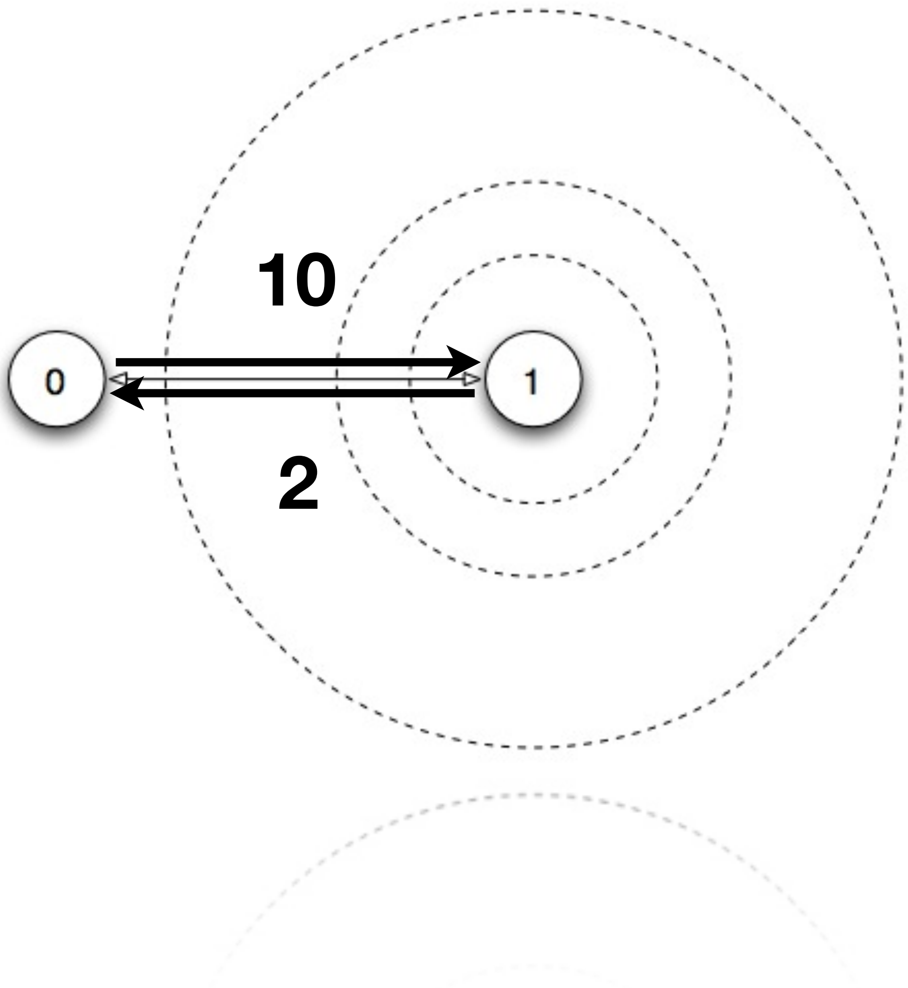
- It is possible that no tour will yield the objective function of the solution framework
- Let the cost of (0,1) be 10 and the cost of (1,0) be 2
- Let the deadhead costs be 1



Plowing with Precedence

Solution Methodology - Solution Framework

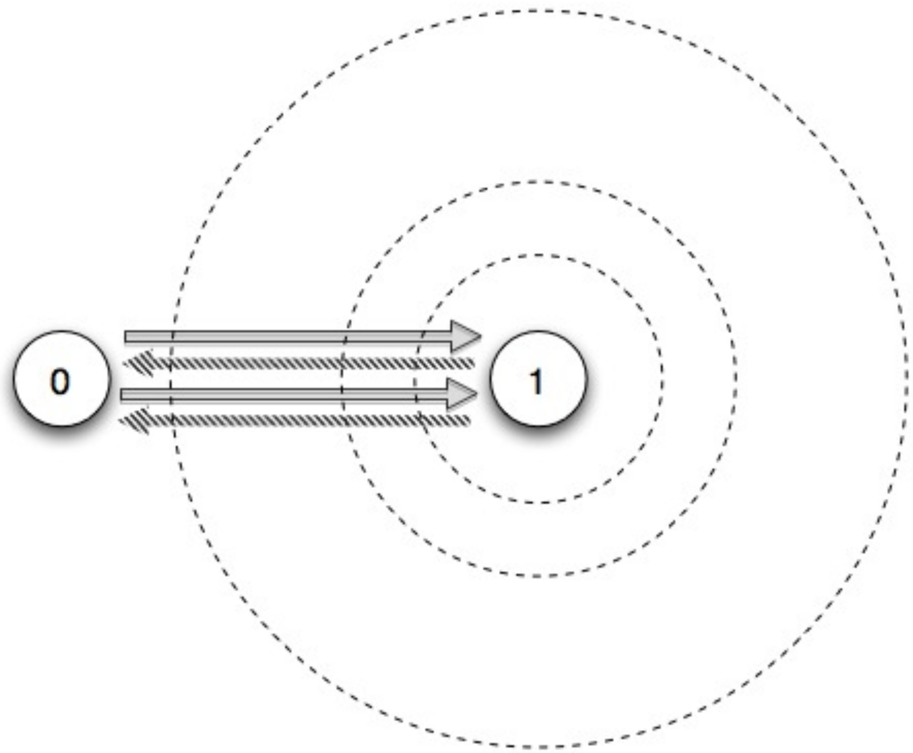
- ❖ It is possible that no tour will yield the objective function of the solution framework
- ❖ Let the cost of (0,1) be 10 and the cost of (1,0) be 2
- ❖ Let the deadhead costs be 1



Plowing with Precedence

Solution Methodology - Solution Framework

- ❖ It is possible that no tour will yield the objective function of the solution framework
- ❖ Let the cost of (0,1) be 10 and the cost of (1,0) be 2
- ❖ Let the deadhead costs be 1

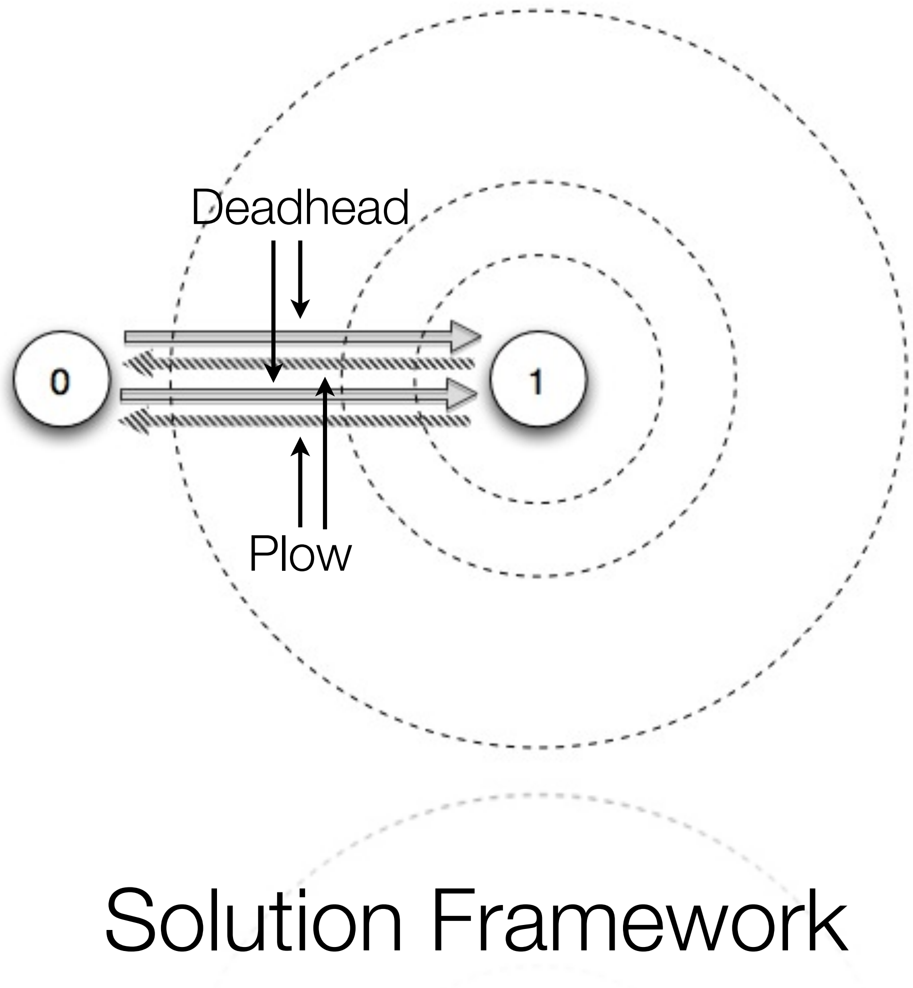


Solution Framework

Plowing with Precedence

Solution Methodology - Solution Framework

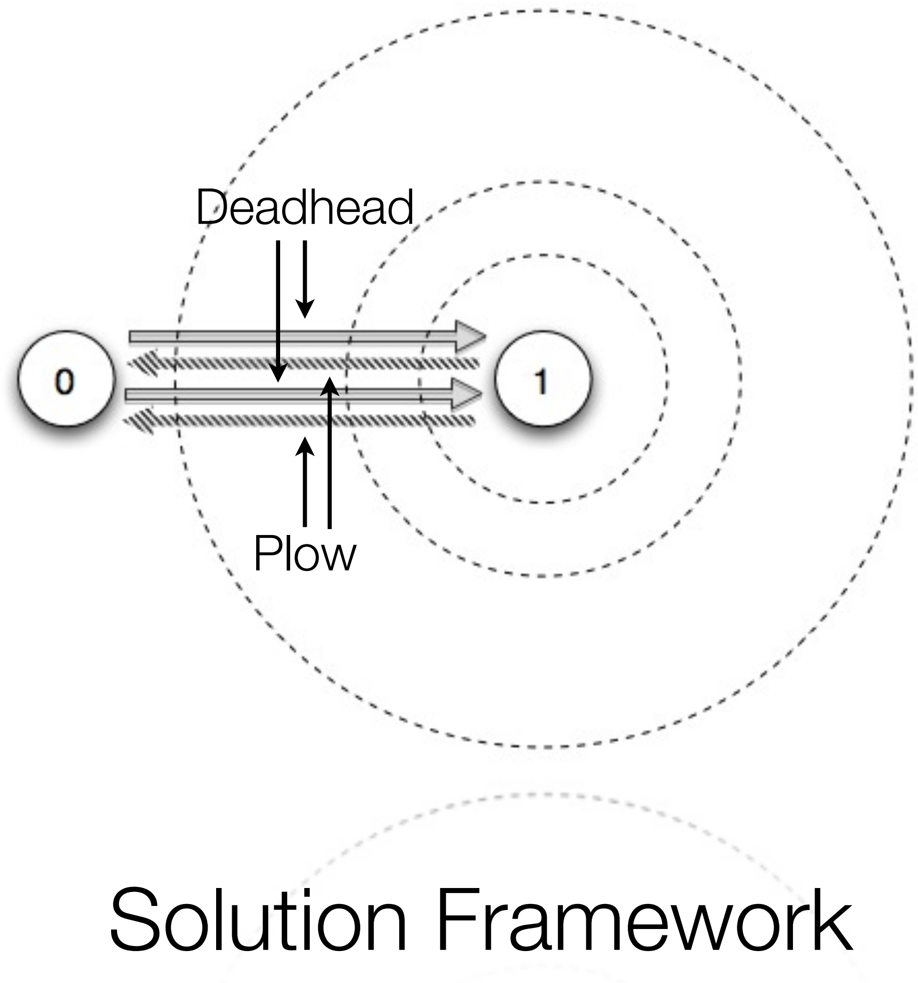
- ❖ It is possible that no tour will yield the objective function of the solution framework
- ❖ Let the cost of (0,1) be 10 and the cost of (1,0) be 2
- ❖ Let the deadhead costs be 1



Solution Framework

Plowing with Precedence

Solution Methodology - Solution Framework



Solution Framework

Plowing with Precedence

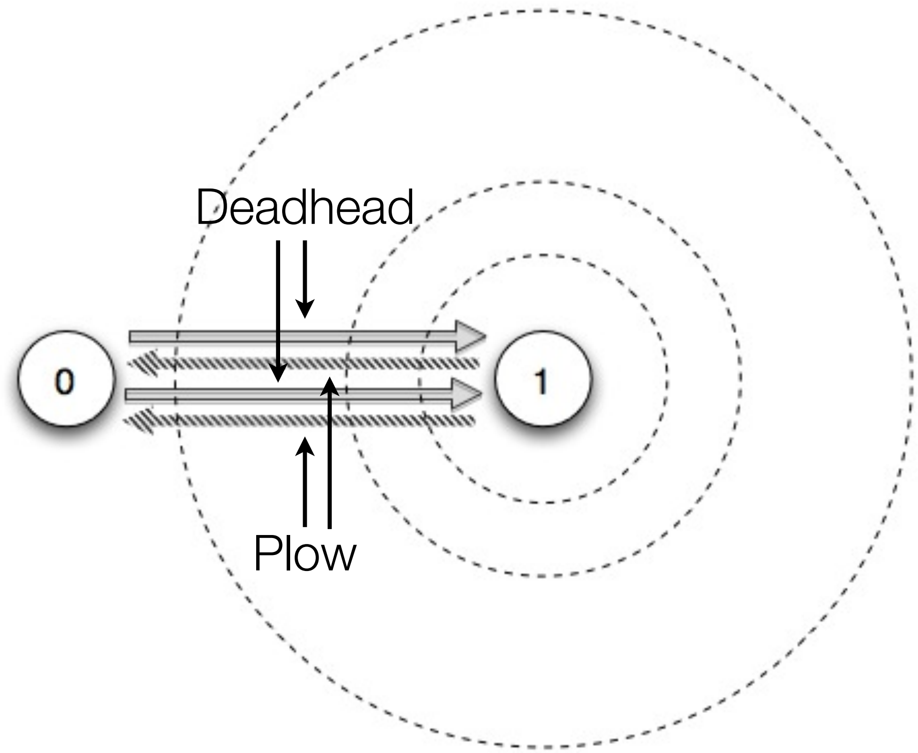
Solution Methodology - Solution Framework

Solution framework seeks to plow downhill twice

Plowing uphill is unavoidable

Solution framework has objective value of 6

Optimal tour {0,1,0} has cost 12



Solution Framework

Plowing with Precedence

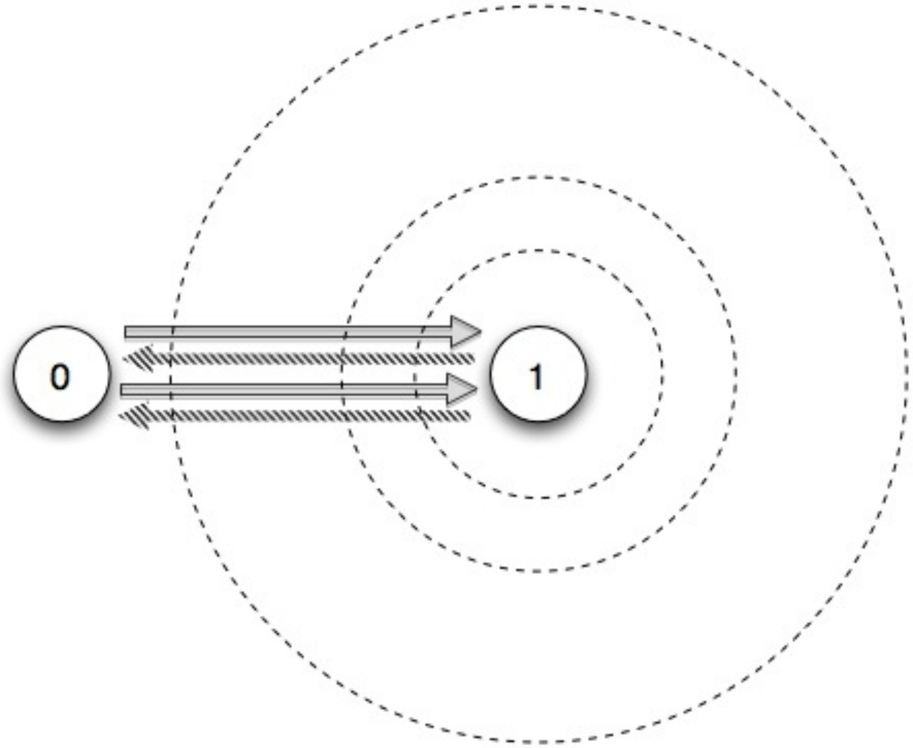
Solution Methodology - Solution Framework

Solution framework seeks to plow downhill twice

Plowing uphill is unavoidable

Solution framework has objective value of 6

Optimal tour {0,1,0} has cost 12



Solution Framework

Plowing with Precedence

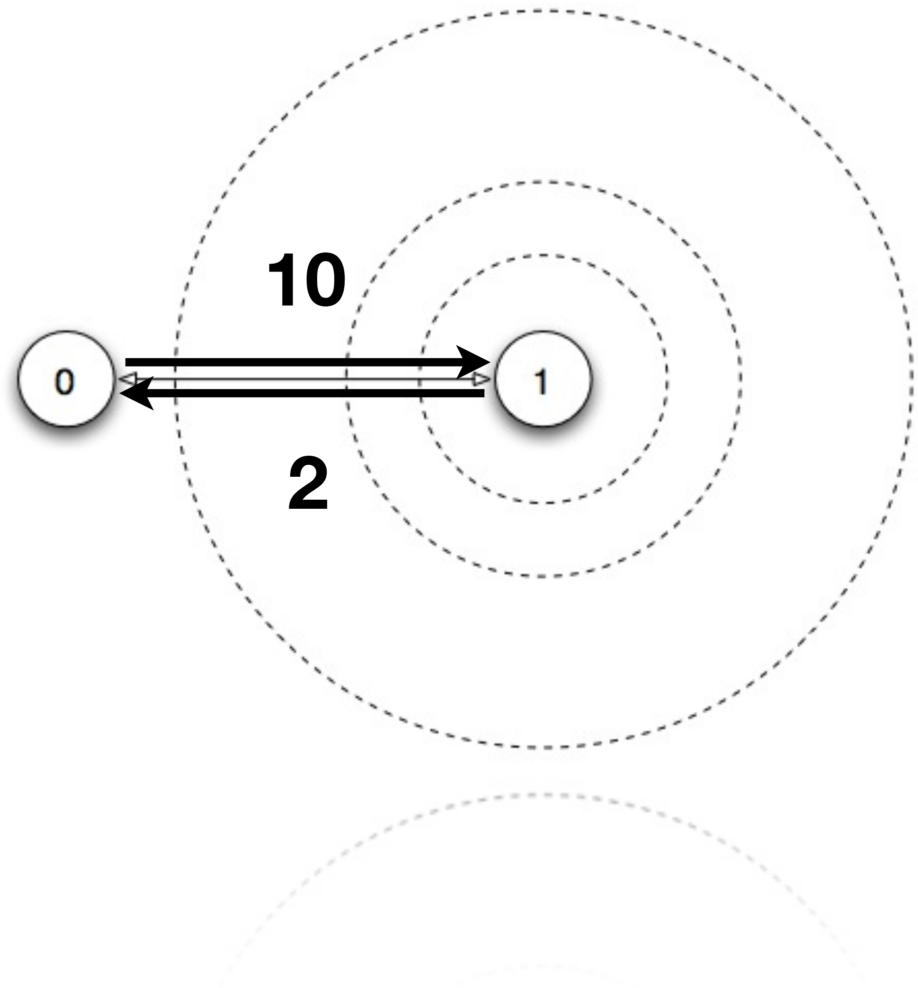
Solution Methodology - Solution Framework

Solution framework seeks to plow downhill twice

Plowing uphill is unavoidable

Solution framework has objective value of 6

Optimal tour {0,1,0} has cost 12



Plowing with Precedence

Solution Methodology - Initial Solution

- A tour can be obtained from solution framework by using Fleury's Algorithm
- This tour is guaranteed to traverse (and hence plow) each street twice
- Not guaranteed to have a cost that is the same as the lower bound of the solution framework (previous example)
- Seek to modify tour using a local search heuristic

Plowing with Precedence

Solution Methodology - Local Search

- We seek to explore the set of all Eulerian tours that obey the solution framework
- Local search searches “nearby” tours in an attempt to find a better one
- Requires:
  - Definition of neighborhood - defines what nearby is
  - Fitness function - gives the quality of a tour
    - In our case, the fitness is the cost of the tour

Plowing with Precedence

Solution Methodology - Local Search

- Solution Fitness:
- For each arc, decide to plow based on the following decision tree:

  if arc has been plowed twice
  → then don’t plow
  else if arc hasn’t been plowed at all
  → then plow
  else if going downhill
  → then plow
  else if tour isn’t going downhill later
  → then plow
  else don’t plow

Plowing with Precedence

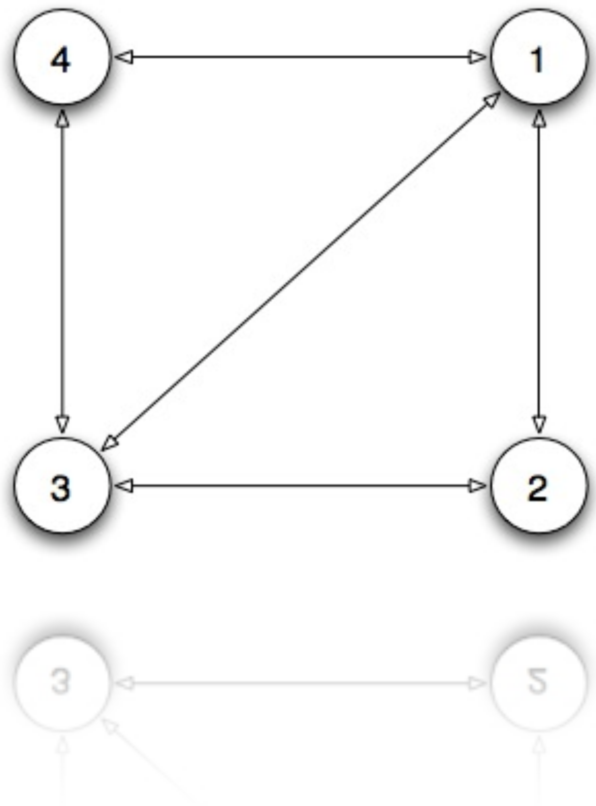
Solution Methodology - Local Search

- All Eulerian tours can be decomposed into cycles
- Definition of neighborhood around a solution s, N(s): the set of all tours that can be obtained by a combination of the following moves:
  - Cycles in the tour are permuted
  - Cycles in the tour are reversed

Plowing with Precedence

Solution Methodology - Local Search

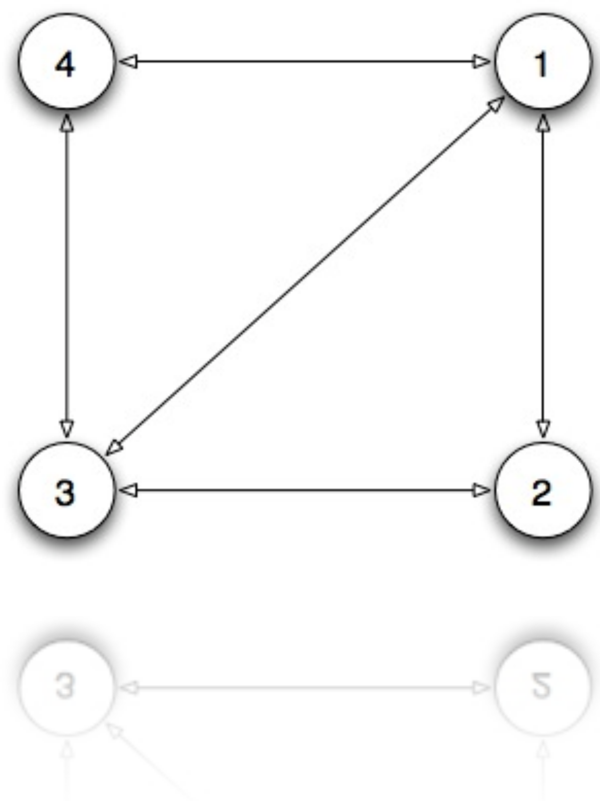
{1,2,3,1,2,3,4,1,3,4,1}



Plowing with Precedence

Solution Methodology - Local Search

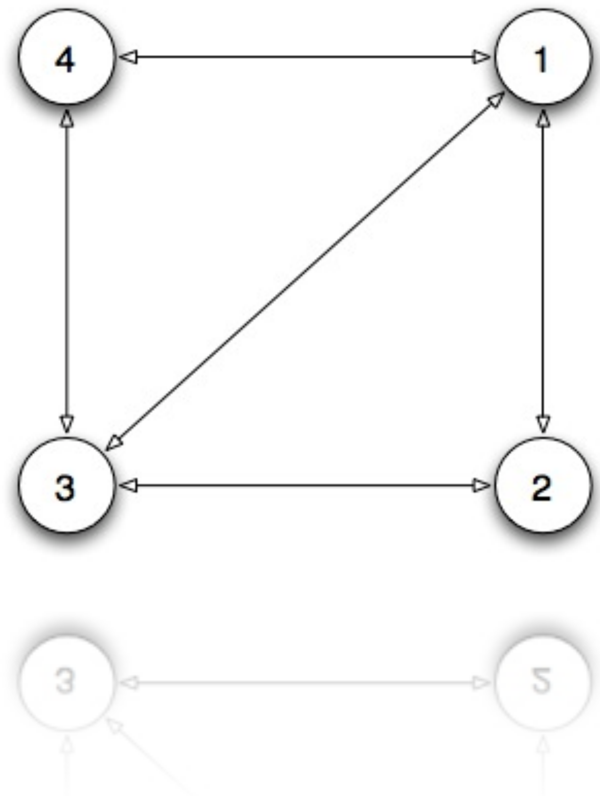
{1,2,3,1,2,3,4,1,3,4,1}



Plowing with Precedence

Solution Methodology - Local Search

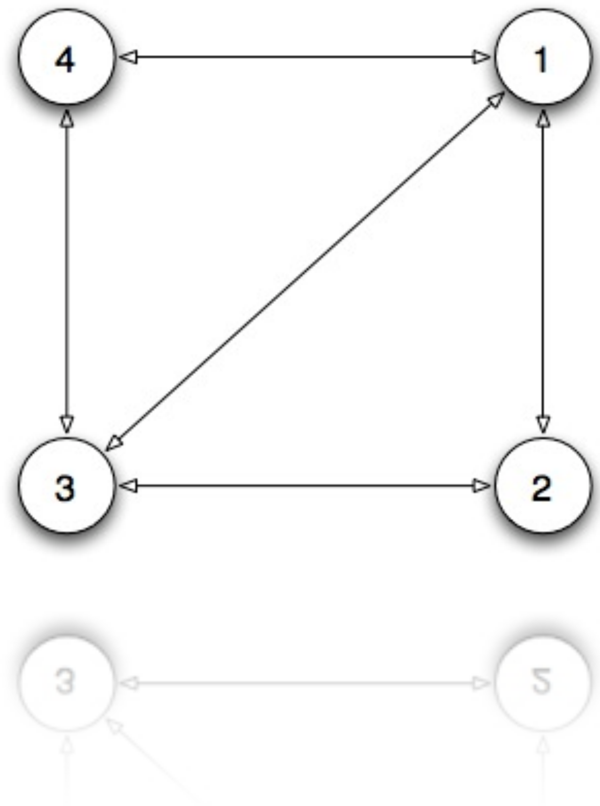
{1,2,3,1,2,3,4,1,3,4,1}



Plowing with Precedence

Solution Methodology - Local Search

{1,2,3,1,2,3,4,1,3,4,1}

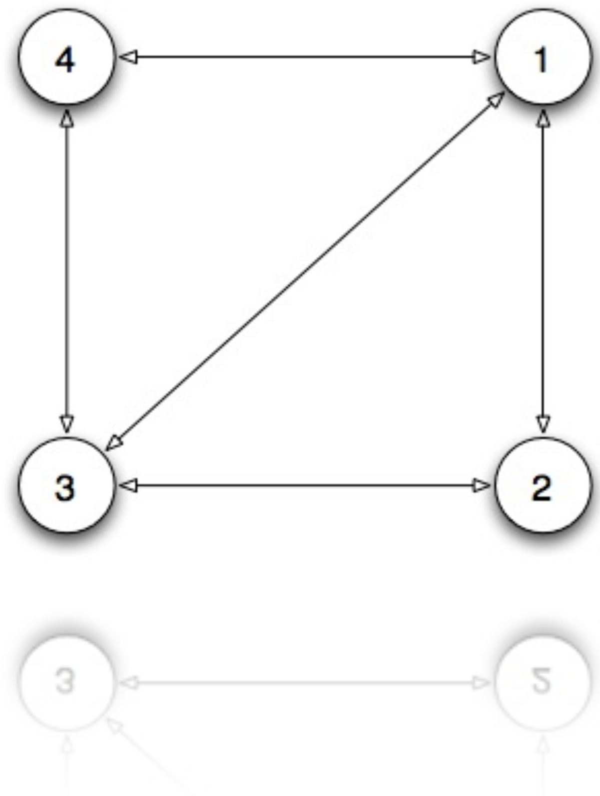


Plowing with Precedence

Solution Methodology - Local Search

{1,2,3,1,2,3,4,1,3,4,1}

{1,2,3,4,1,3,4,1,2,3,1}

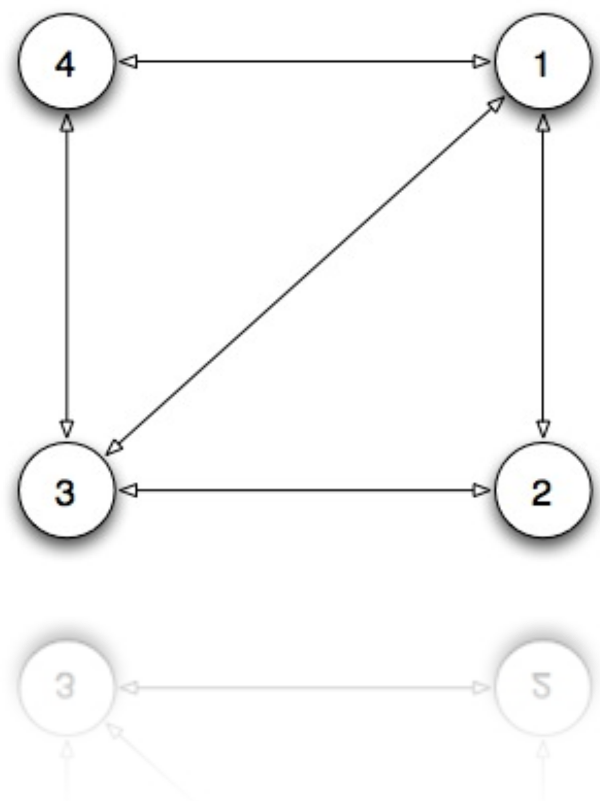


Plowing with Precedence

Solution Methodology - Local Search

{1,2,3,1,2,3,4,1,3,4,1}

{1,2,3,4,1,3,4,1,2,3,1}

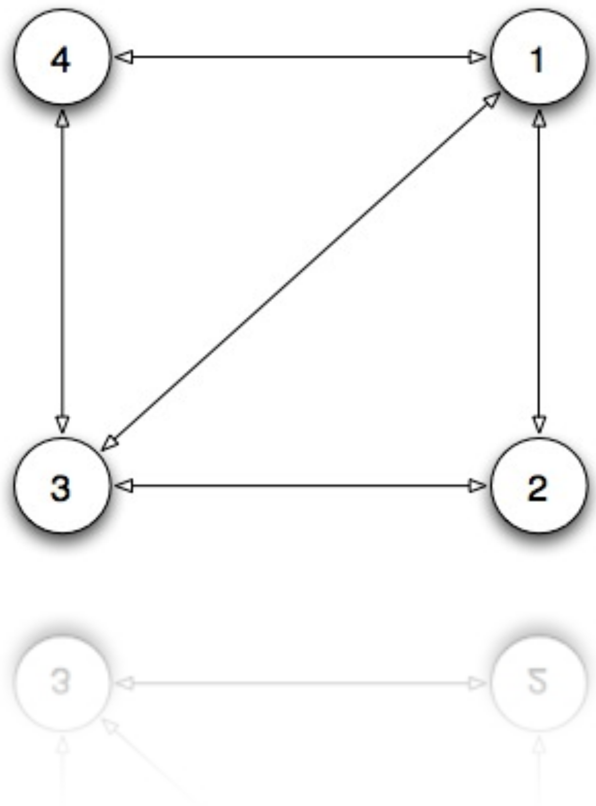


Plowing with Precedence

Solution Methodology - Local Search

{1,2,3,1,2,3,4,1,3,4,1}

{1,2,3,4,1,3,4,1,2,3,1}

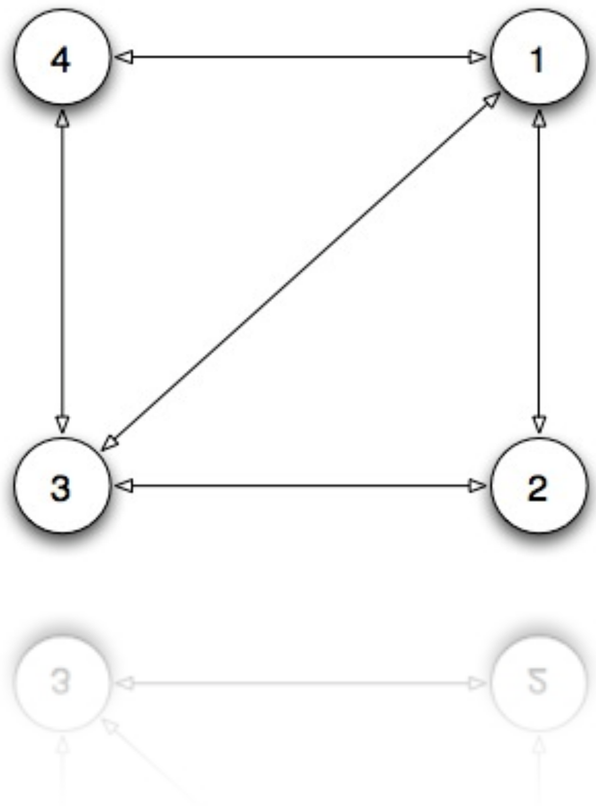


Plowing with Precedence

Solution Methodology - Local Search

{1,2,3,1,2,3,4,1,3,4,1}

{1,2,3,4,1,3,4,1,2,3,1}

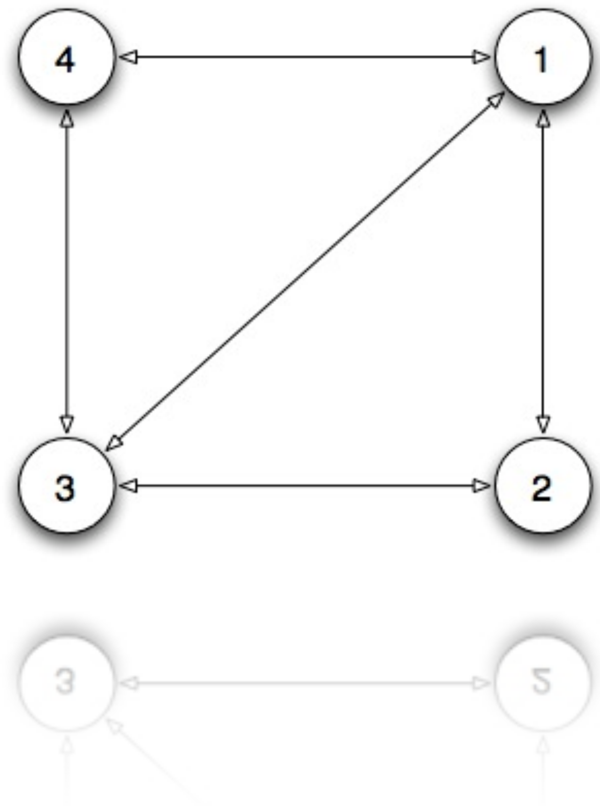


Plowing with Precedence

Solution Methodology - Local Search

{1,2,3,1,2,3,4,1,3,4,1}

{1,2,3,4,1,3,4,1,2,3,1}

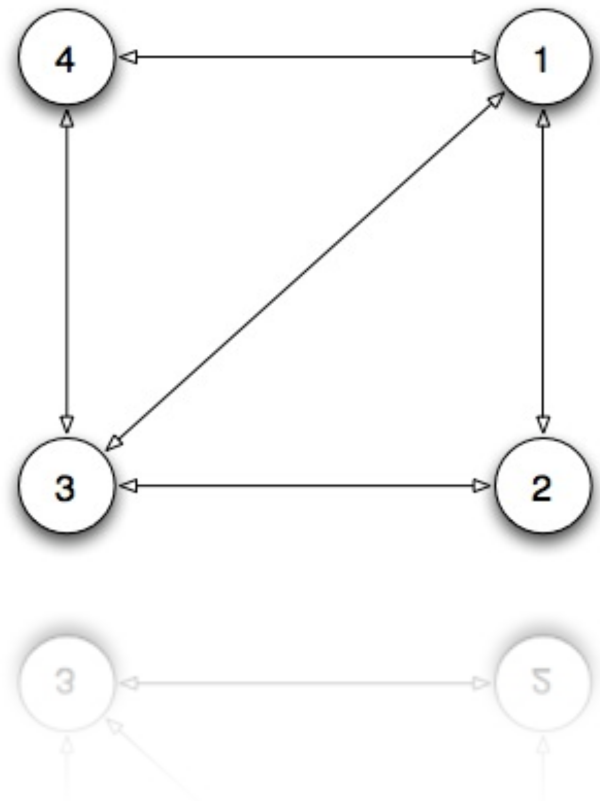


Plowing with Precedence

Solution Methodology - Local Search

{1,2,3,1,2,3,4,1,3,4,1}

{1,2,3,4,1,3,4,1,2,3,1}



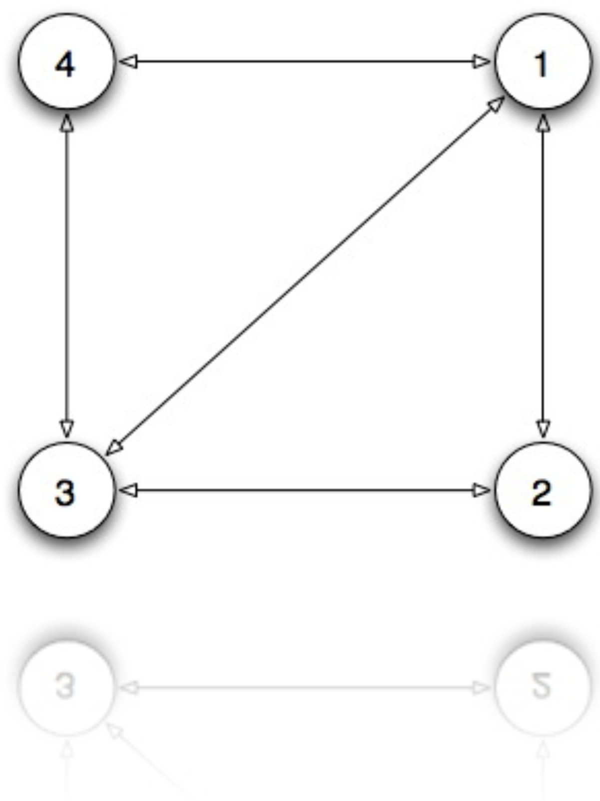
Plowing with Precedence

Solution Methodology - Local Search

{1,2,3,1,2,3,4,1,3,4,1}

{1,2,3,4,1,3,4,1,2,3,1}

{1,2,3,4,1,3,4,1,3,2,1}



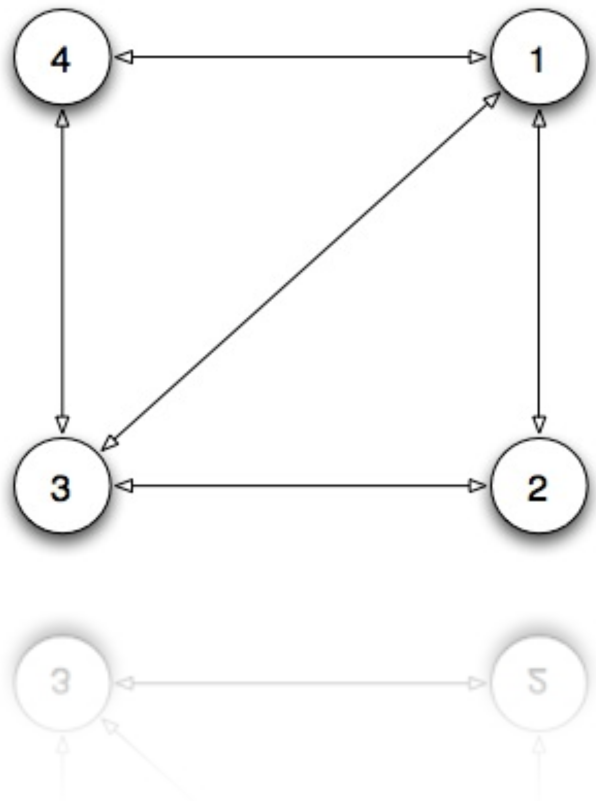
Plowing with Precedence

Solution Methodology - Local Search

{1,2,3,1,2,3,4,1,3,4,1}

{1,2,3,4,1,3,4,1,2,3,1}

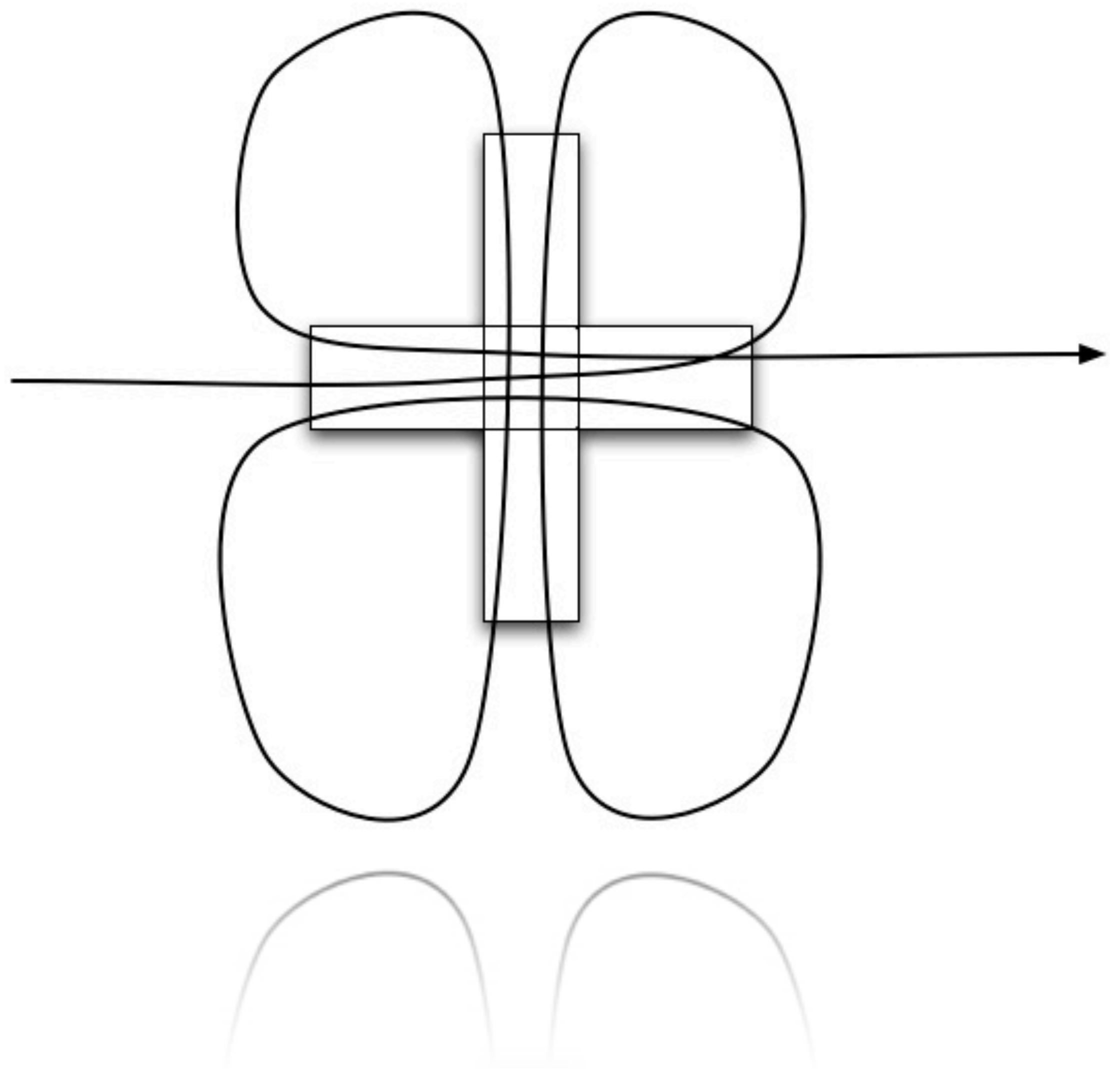
{1,2,3,4,1,3,4,1,3,2,1}



Plowing with Precedence

Solution Methodology - Local Search

- The number of permutations is large: n! for n cycles
- To limit the size of the neighborhood, if n>4, we limit the set of permutations to 4!+n for linear growth
- Most intersections have four or fewer cycles



Plowing with Precedence

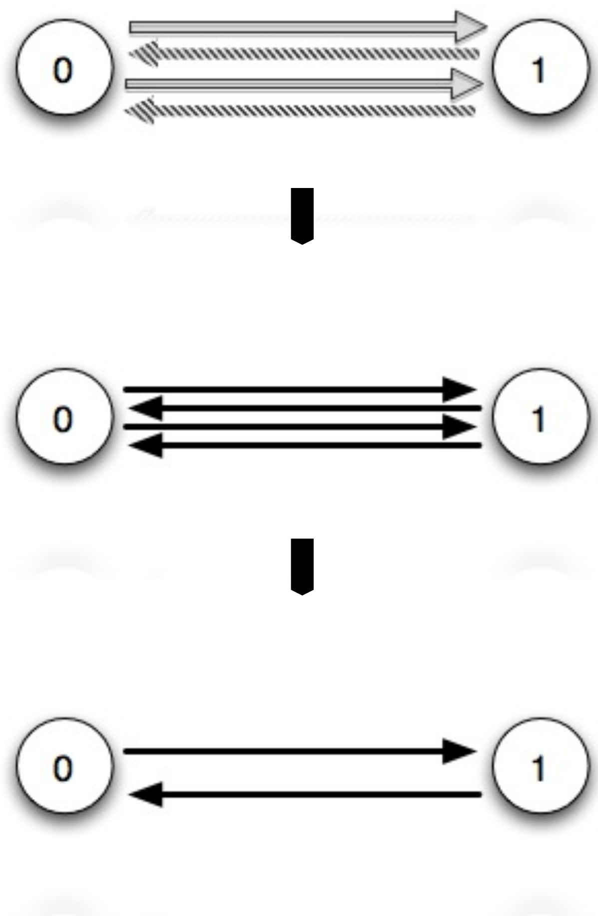
Solution Methodology - Reinitialization

- Local search is deterministic and dependent on the initial solution
- We reinitialize to obtain new initial solutions to run the local search procedure on
- This is done by permuting cycles around different nodes randomly a large number of times
- The best solution obtained by 15 runs of the local search and reinitialization combination is retained

Plowing with Precedence

Solution Methodology - Pruning

- It is possible that a tour will have cycles that consist of entirely deadhead
- These cycles can be pruned to obtain a lower-cost tour that still plows each street twice
- Pruning is done at the end of local search + reinitialization phase



Plowing with Precedence

Results

- We test our algorithm on a bank of modified Windy Rural Postman Problems presented in Corberan et al.
  - Remove Rural concept
  - Existing costs are interpreted as plowing costs
  - Randomly generate deadhead costs
- Instances are characterized by:
  - Number of nodes (7-196)
  - Number of arcs (10-316)
  - Average cost deviation - average discrepancy in cost between plowing up and plowing down (4%-80%)

Plowing with Precedence

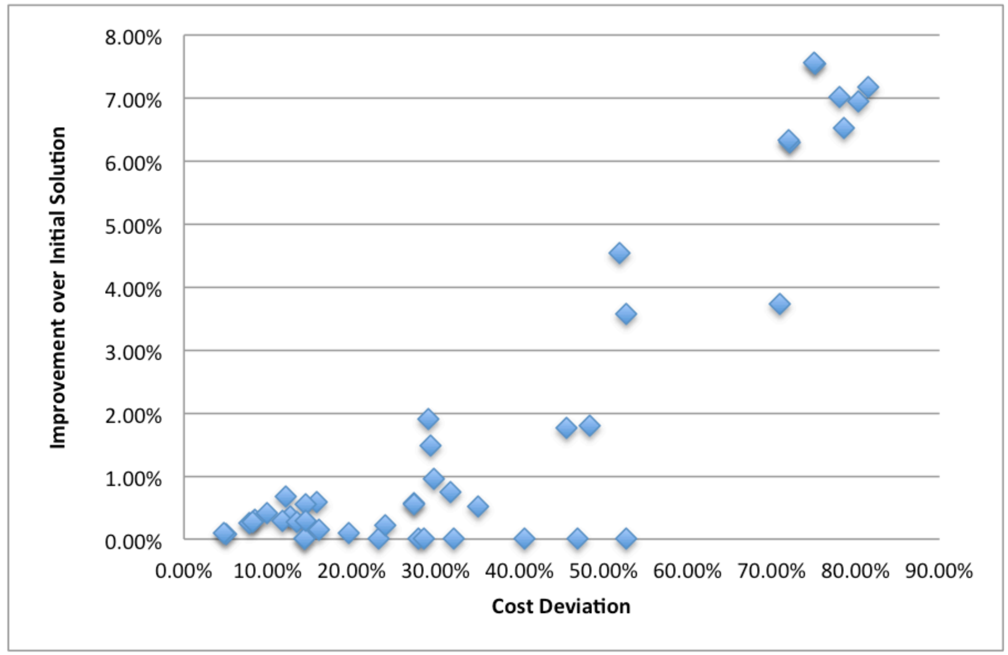
Results

- Our IP formulation is too large to solve all but the smallest of instances
- We therefore compare against the lower bound given by the solution framework
  - If we obtain the lower bound, then we know we have the optimal solution
- Our algorithm performs very well
  - Obtains optimal solution on more than 50% of the instances
  - Averages 0.2% deviation from the lower bound

Plowing with Precedence

Results

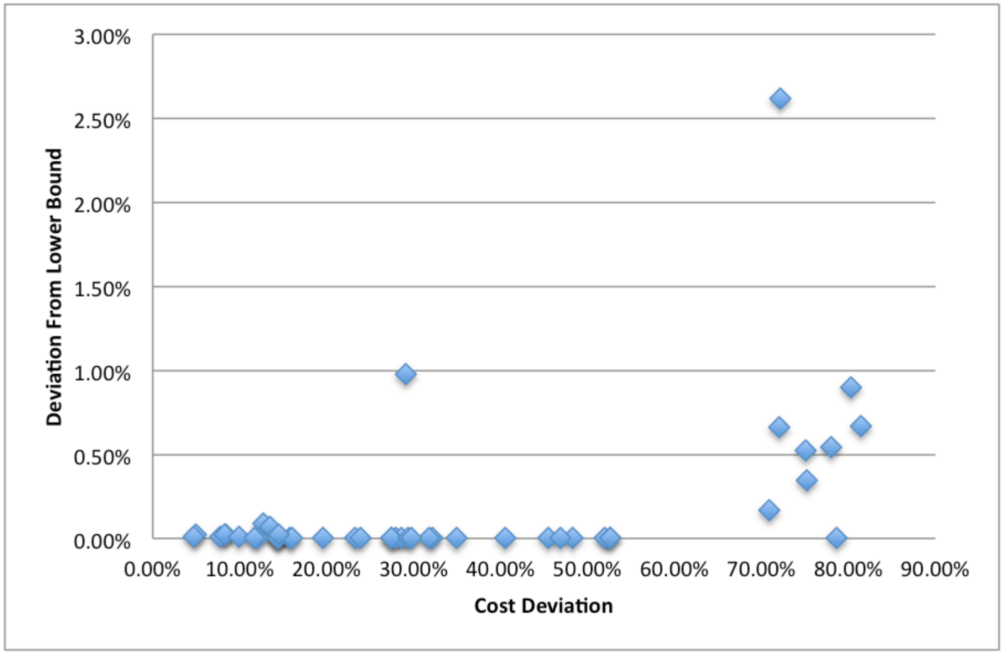
- Compare final solution cost against the initial solution cost
- 1.8% average improvement
- Measure percentage improvement vs. Average cost deviation



Plowing with Precedence

Results

- Cost deviation is largest driving factor in deviation from lower bound
- 0.17% average deviation from the lower bound
- Deviation from LB increases as cost deviation increases
- Want to investigate further



Plowing with Precedence

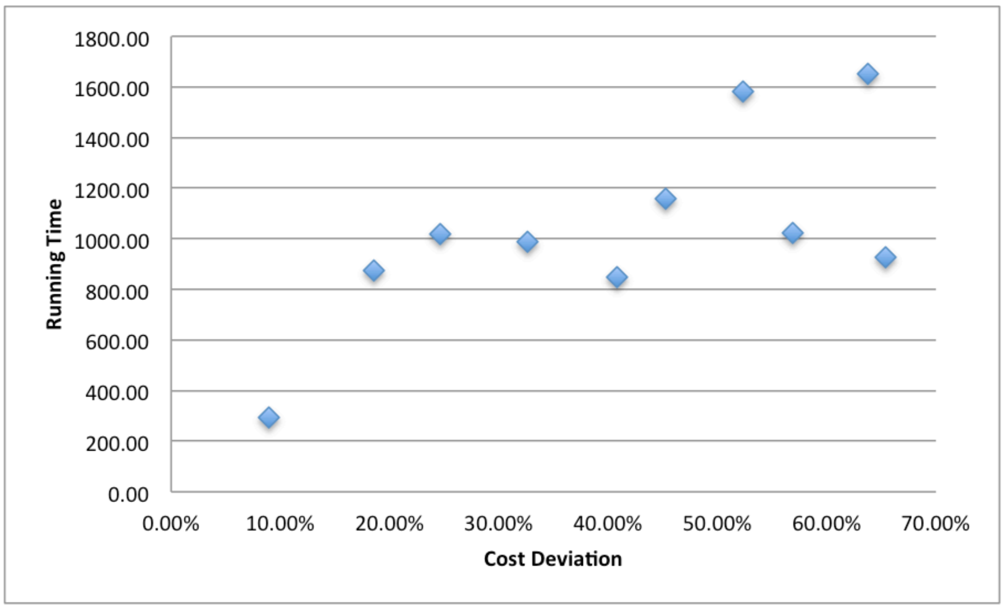
Results

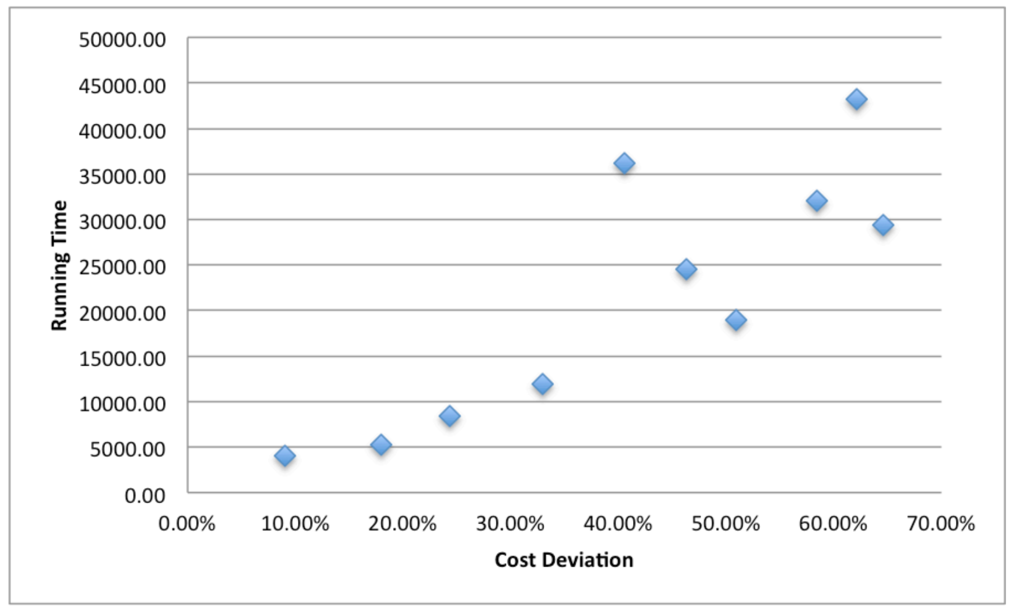
- We took two large instances and constructed several new instances that:
  - Preserved the same graph
  - Average cost deviation ranged from 10% to 70%
- Compare the effects of average cost deviation on:
  - Running Time
  - Percentage Improvement
  - Deviation from Lower Bound

Plowing with Precedence

Results

Running Time vs. Average Cost Deviation





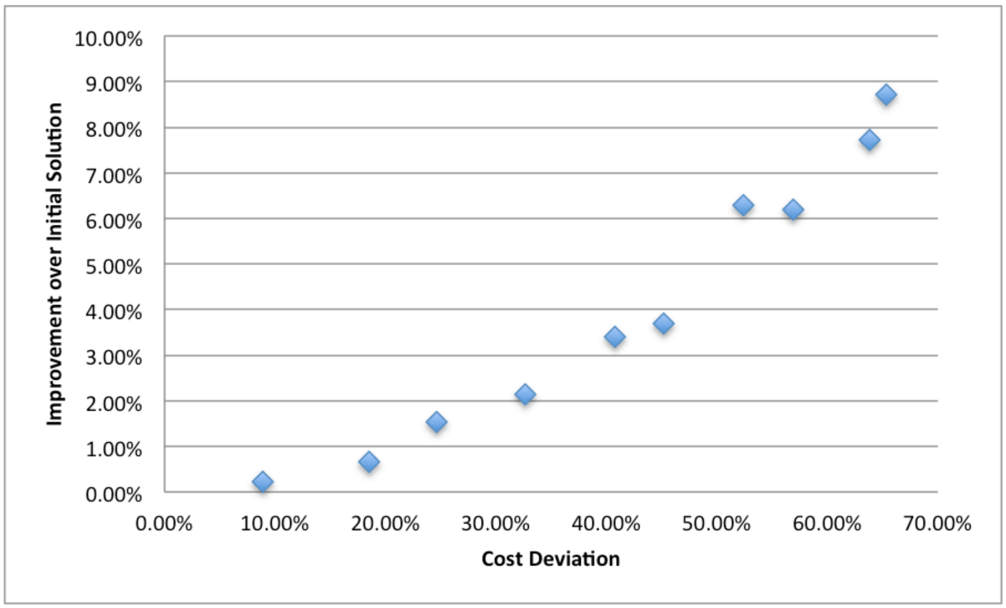
Instance A3101

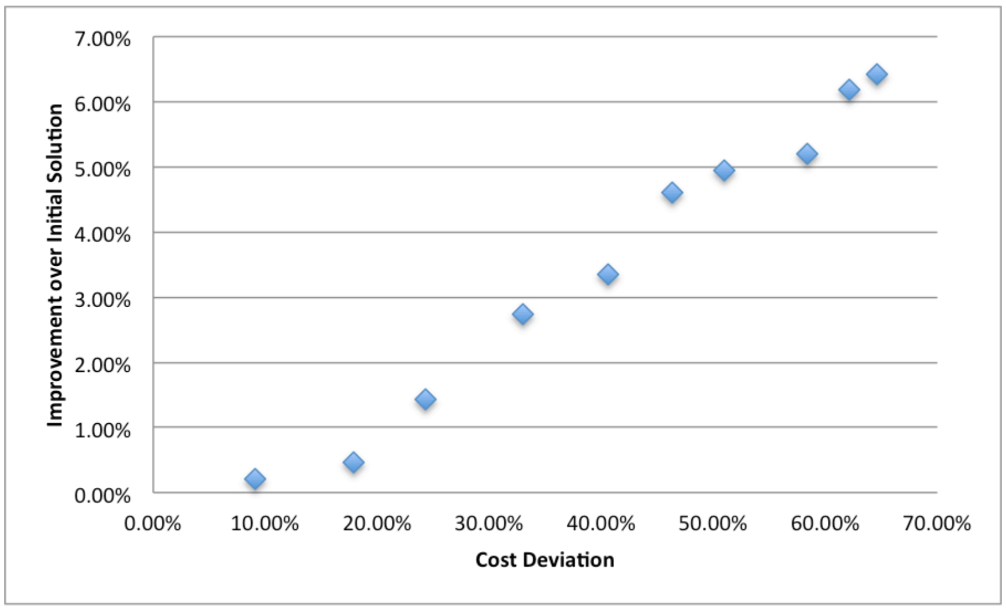
Instance M3101

Plowing with Precedence

Results

Percentage Improvement vs. Average Cost Deviation





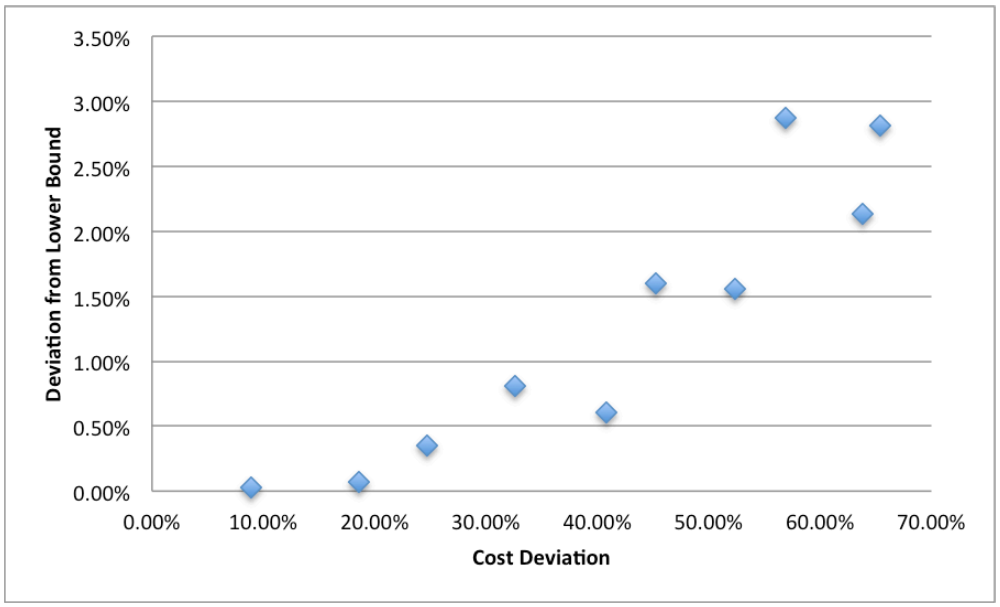
Instance A3101

Instance M3101

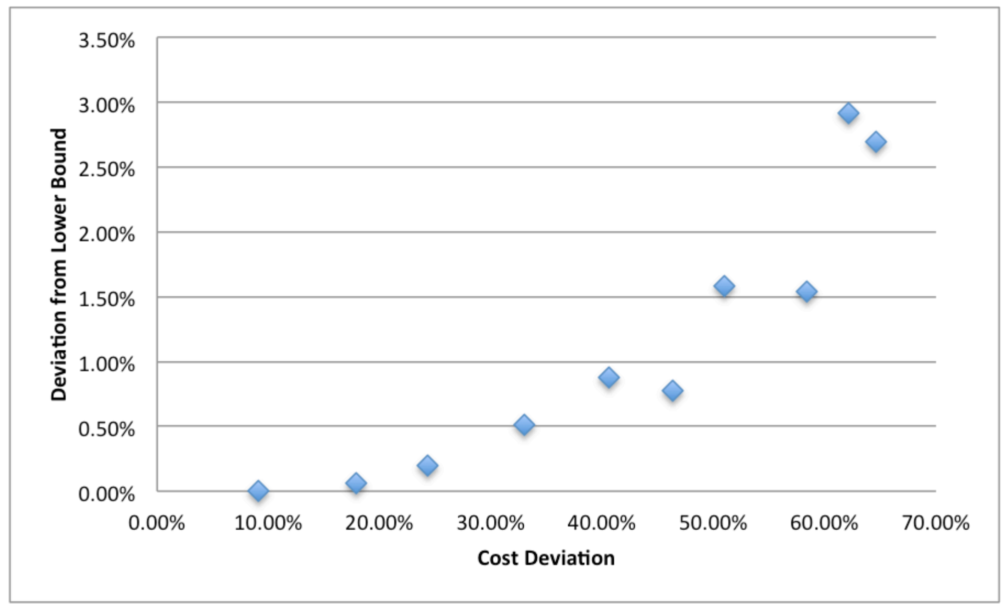
Plowing with Precedence

Results

Deviation from Lower Bound vs. Average Cost Deviation



Instance A3101



Instance M3101

Plowing with Precedence

Conclusions

- Introduced a variant of the WPP
- Addresses the practical consideration that deadheading a street after plowing is less costly than plowing the street
- Introduces the concept of precedence to postman problems
- Obtain very good results, producing solutions that are, on average, within 0.79% of the lower bound
  - Solutions are very often optimal
- Observed increases in running time, percentage improvement, and deviation from the lower bound as a function of the average cost deviation

Plowing with Precedence

Conclusions

- Future work:
  - Generalize the concept of precedence: Let the cost of traversal be a general function of the number of times traversed
  - Add multiple plows: When one plow plows a street, other plows are able to deadhead that street