

STOCHASTIC LOCAL SEARCH FOUNDATIONS & APPLICATIONS

Travelling Salesman Problems

Presented by Camilo Rostoker
rostokec@cs.ubc.ca
Department of Computer Science
University of British Columbia





Outline

1. Introduction to the TSP
2. Benchmarks
3. State-of-the-art methods overview
4. Construction Heuristics
5. The Lin-Kernighan Algorithm and variants
6. ILS Algorithms for the TSP
7. Population-based Algorithms for the TSP
8. Further readings & related work
9. Conclusion & summary



Problem Definition

- n Given an edge-weighted, completely connected, directed graph $G = (V, E, w)$ with:

V is set of $n = \#V$ vertices

E is the set of directed edges

$w : E \rightarrow \mathbb{R}^+$ is a function assigning each $e \in E$ a weight $w(e)$

- n Goal: find a minimum weight Hamilton cycle in G



Why the TSP?

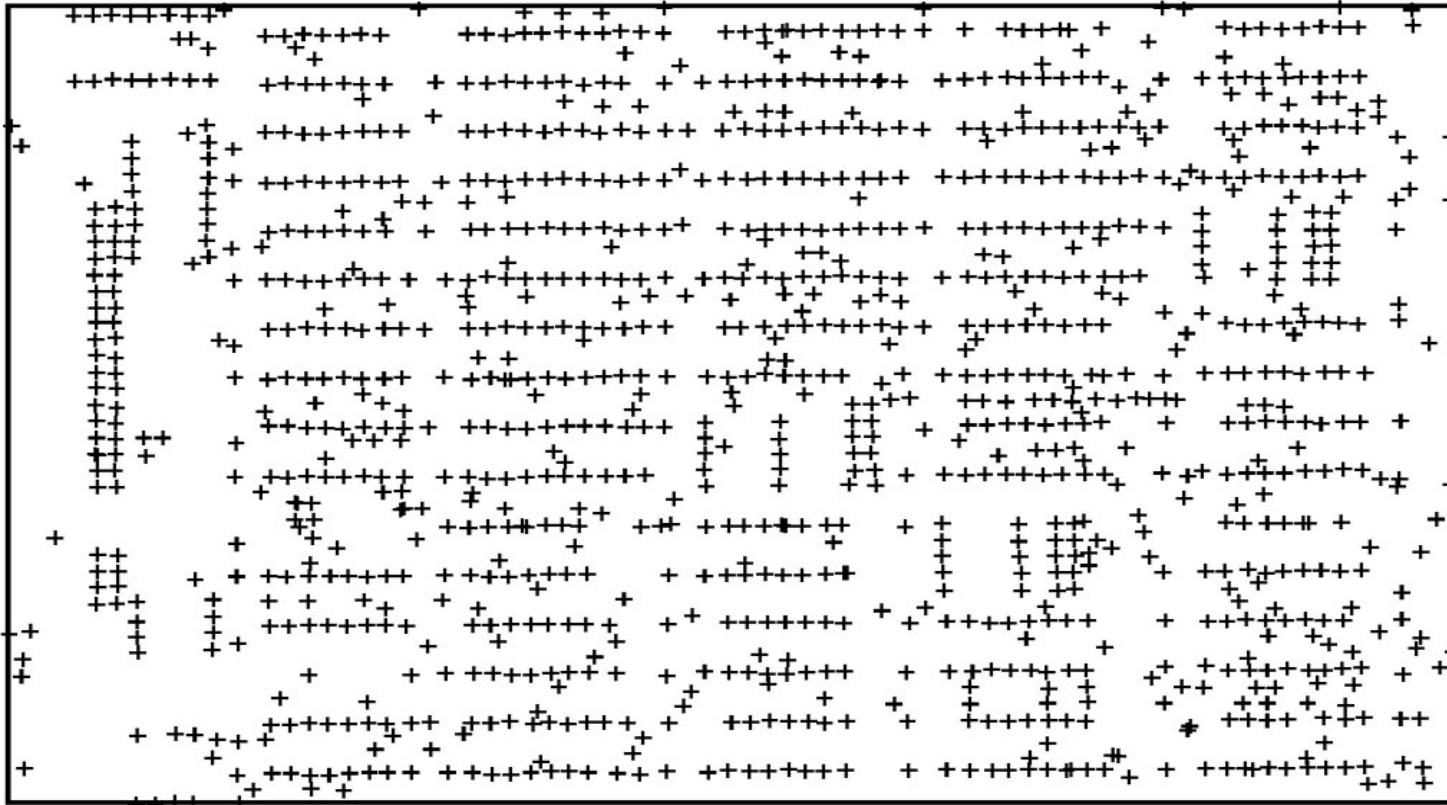
- n Important for both research and applications
- n Most combinatorial optimization algorithms have been developed using TSP
- n Conceptually simple à easily understood and explained
- n NP-hard à good for theoretical algorithmics



TSP Benchmark Instances

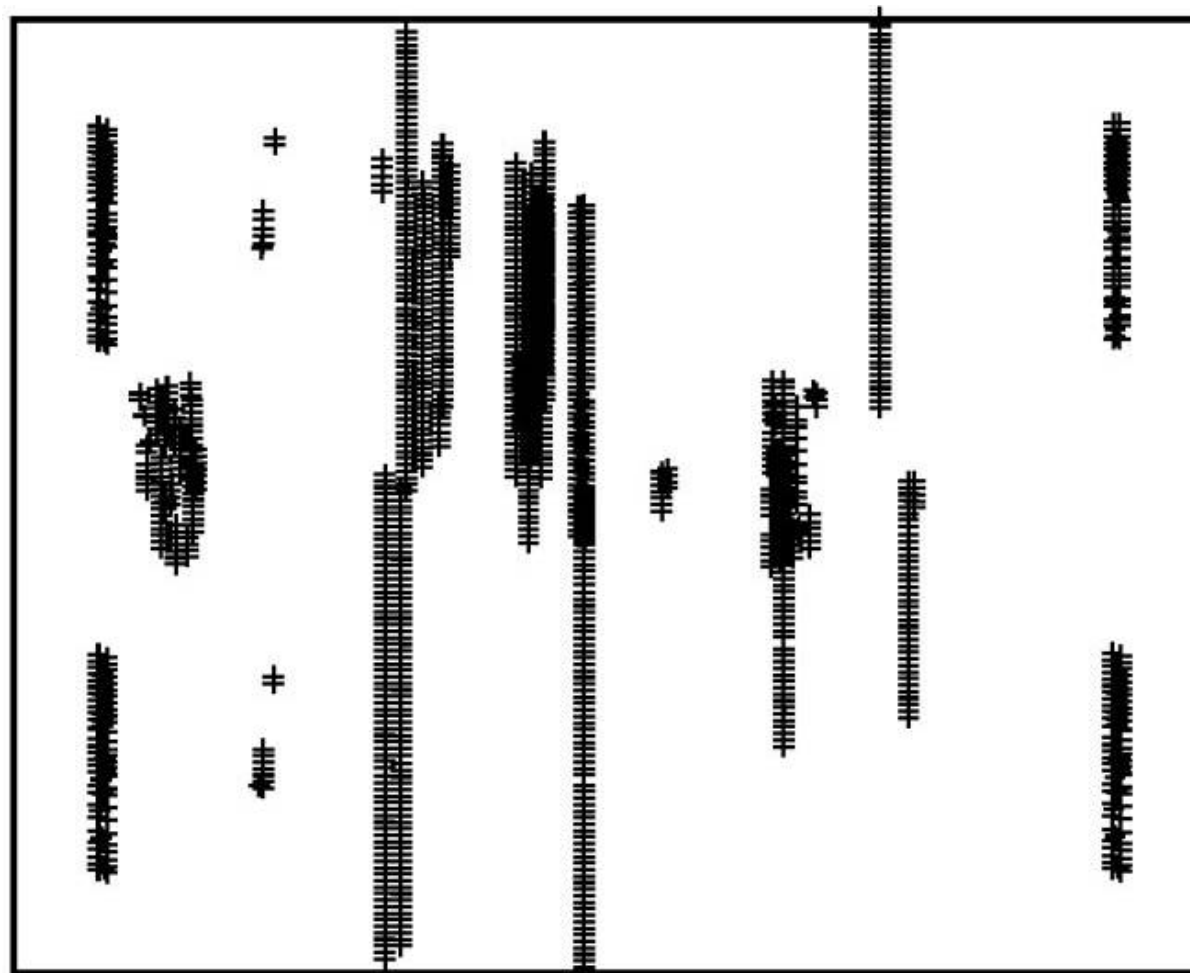
- n TSPLIB – benchmark library for TSP
 - .. More than 100 instances, up to 85,900 cities
 - .. Problem instances from circuit board drilling, X-ray crystallography, geo-spatial datasets, beer garden optimization problems (BGOP) & more!
- n Random generated instances have been a key to experimental and theoretical work
 - .. Random Distance Matrix (RDM)
 - .. Random Uniform Euclidean (RUE)
 - .. Random Clustered Euclidean (RCE)

Circuit Board Drilling



TSPLIB instance pcb1173

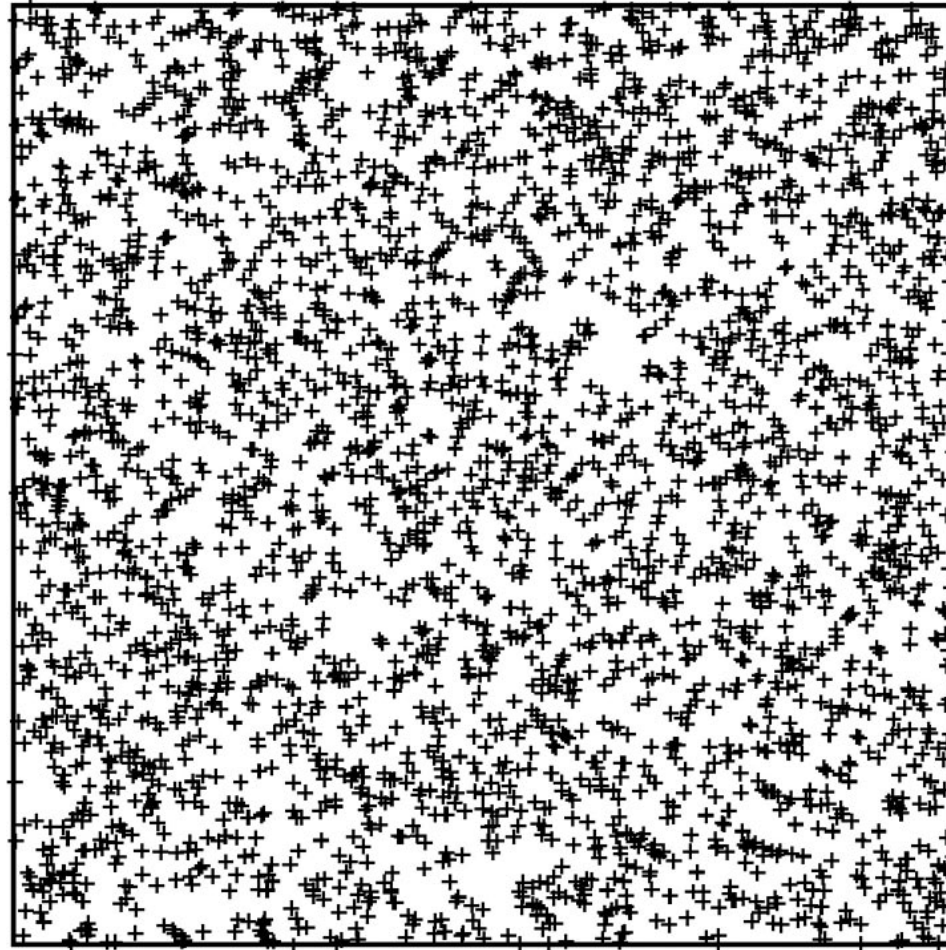
Circuit Board Drilling



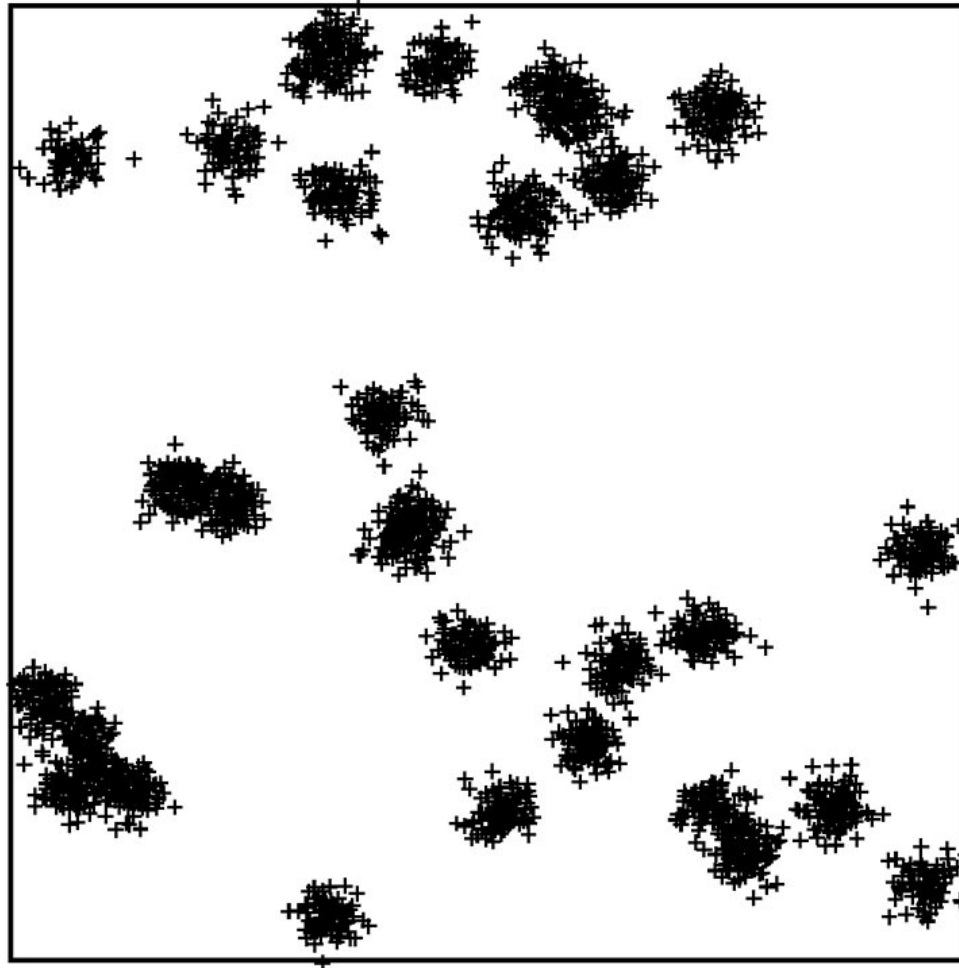
TSPLIB instance pcb1173



Random Uniform Euclidean (RUE)



Random Clustered Euclidean





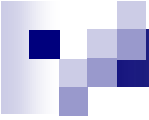
Symmetric vs. Asymmetric TSP

- n Symmetric TSP instances have $w(v,v') = w(v',v)$ for all (v,v') in V
- n Asymmetric TSP (ATSP) instances have at least one pair of vertices (v,v') such that $w(v,v') \neq w(v',v)$
- n ATSP are typically harder to solve
- n Real world ATSP problems include
 - .. Moving drills along a tilted surface
 - .. Scheduling read operations on a computer disk
 - .. Collecting coins from a payphone
 - .. Genome reconstruction
- n How to solve ATSP?
 - .. Use native ATSP algorithms
 - .. Transform ATSP into symmetric TSP
- n For this presentation assume Symmetric



Lower bounds on the TSP

- n Lower bounds are necessary to rate quality guarantees of tours
- n Important for study of complete algorithm
- n Use simple relaxation method to obtain rough lower bound
- n Best lower bound: Held-Karp bounds
 - .. experimentally, shown to be very tight
 - .. within less than 1% of optimum for random Euclidean
 - .. up to 2% for TSPLIB instances



State-of-the-Art Methods

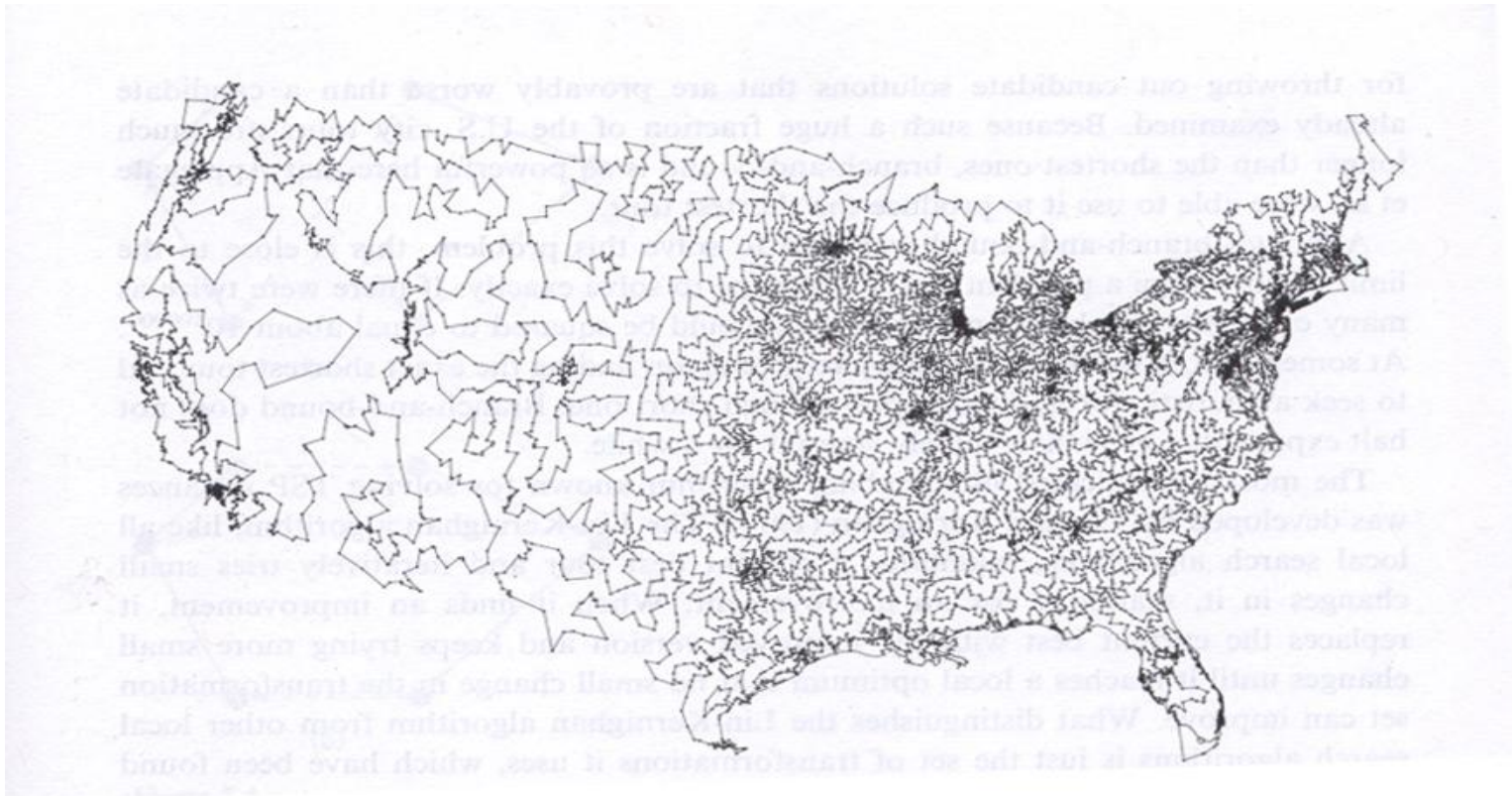
n Complete algorithms

- .. Most based on branch & cut methods
- .. Current largest solved & proved instance is 24,978 cities
- .. Concorde is a distributed TSP solving environment
 - n Solved 106 out of the 110 TSPLIB instances

n SLS algorithms

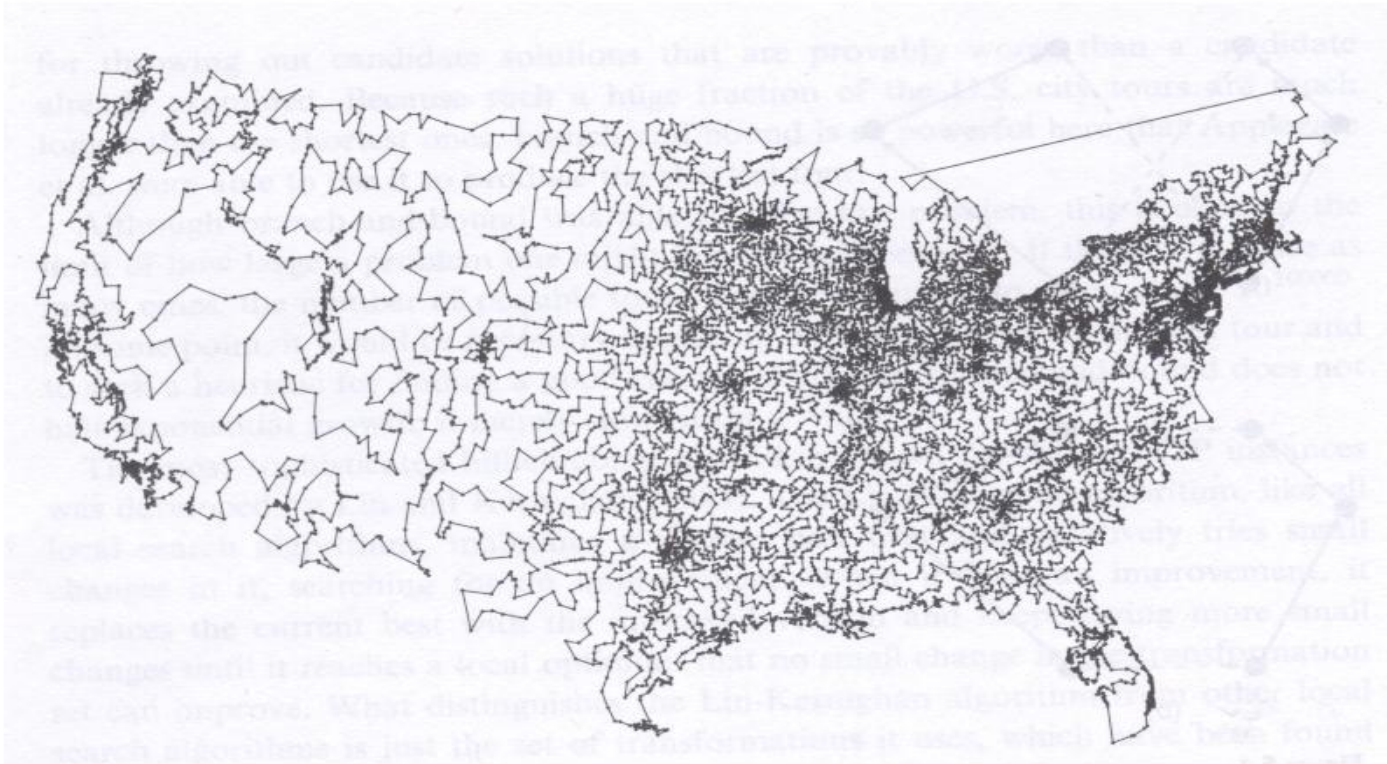
- .. Enables us to specify degree of tradeoff between accuracy and computation time
- .. Construction heuristics
- .. Iterative improvement methods
- .. Hybrid methods
- .. Population-based methods

Complete vs. SLS



TSP solved for the 13, 509 U.S. cities with population over 500

Complete vs. SLS



The LK solution – only 2.3% above the optimum

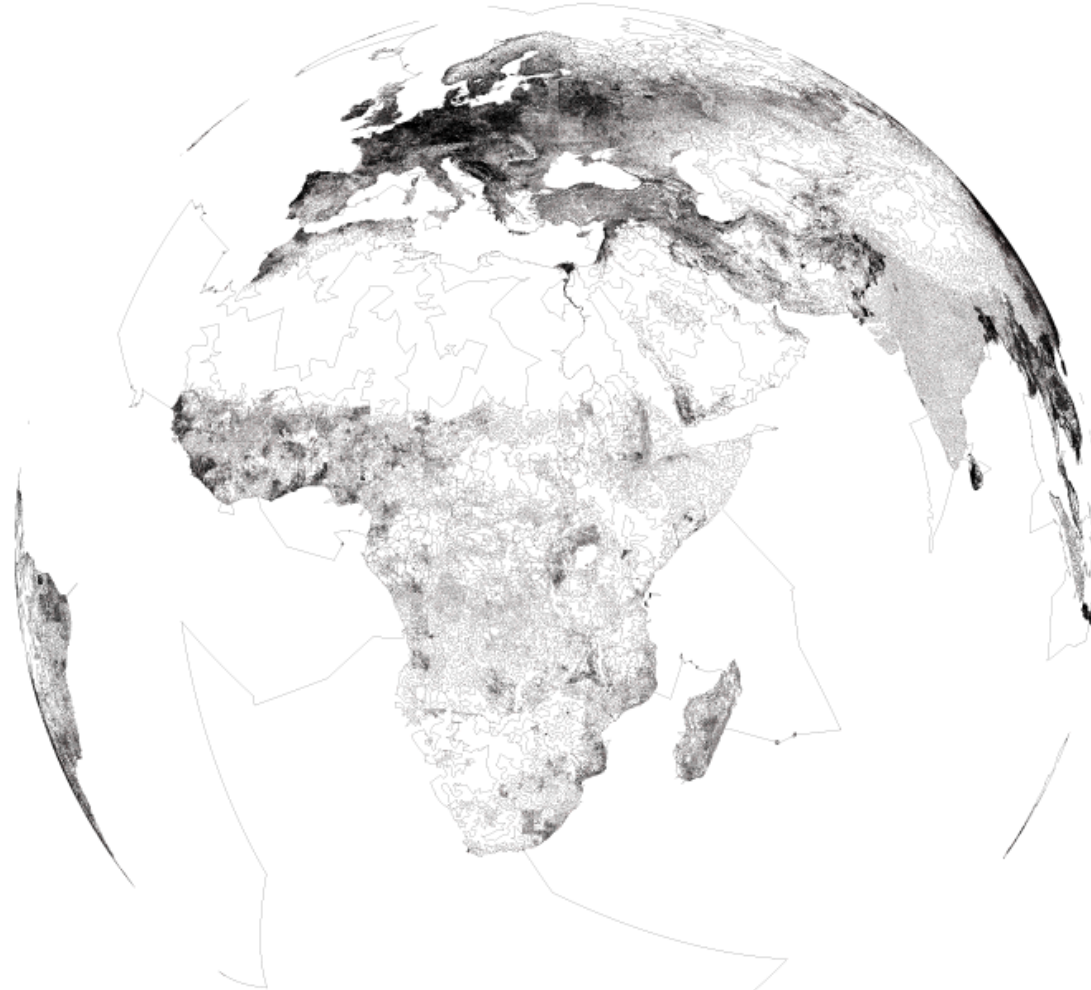
Current Largest Solved TSP

- n TSP for all 24,978 cities in Sweden was solved & proved to be optimal
- n Tour of length 855,597 TSPLIB units (approximately 72,500 kilometers)
- n Used various complete methods and 85 CPU years to prove.
- n Surpassed the previous record of 15,112 cities through Germany set in April 2001.
- n Algorithm by Helsgaun using a version of his LK-H code.





What's Next?



World TSP - 1,504,711 cities



Construction Heuristics

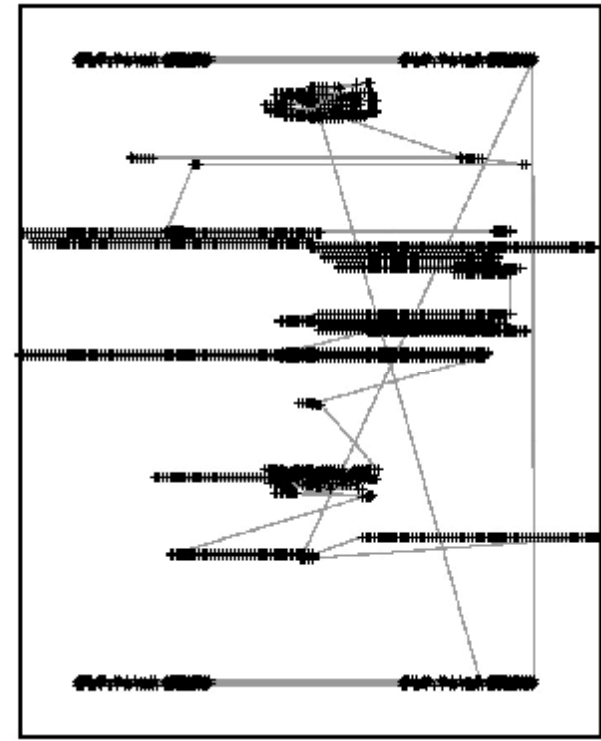
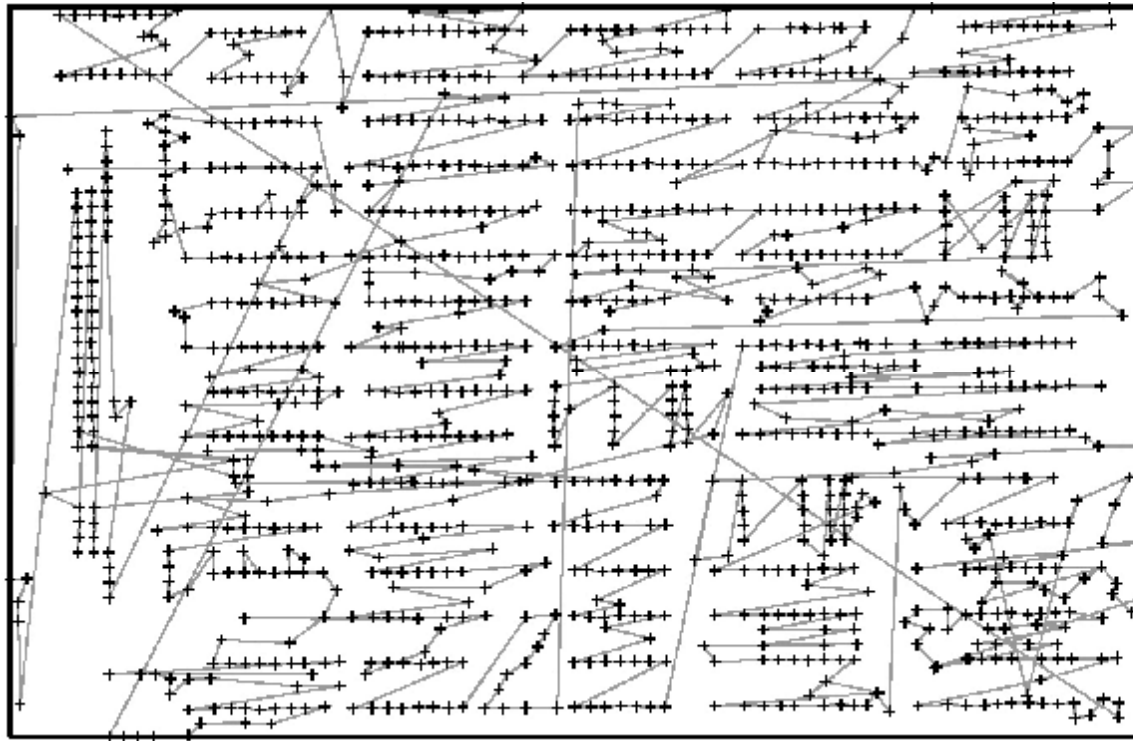
- n Simple SLS methods that can quickly construct reasonably good tours
- n Often used as initialization procedures for more advanced SLS algorithms
- n Types of construction heuristics:
 - .. Iteratively extend a connected partial tour
 - .. Build up tour fragments and piece them together in the end
 - .. Minimum spanning tree (MST)



Nearest-Neighbour Heuristic (NNH)

- n Start with randomly chosen vertex
- n Iteratively extend current partial tour by adding an unvisited vertex connected by a minimum-weight edge
- n Tours are usually locally similar to optimal solution but contain a few very long edges

Results for Nearest-Neighbour Tours





Insertion Construction Heuristic

- n In each step, extend partial tour p by inserting a vertex v that leads to minimal increase
- n Types of insertion heuristics
 - Nearest Insertion
 - Cheapest Insertion
 - Farthest Insertion
 - Random Insertion
- n Nearest and cheapest and provably at most twice as long as optimal tour
- n Random/farthest only guaranteed to be within $O(\log n)$ of optimal



Fragmented Construction Heuristics

- n Build up multiple tour fragments and piece them together to form a complete tour

- n Greedy Heuristics
 - .. Sort all edges in G by weight, then begin linear insertion of edges

- n Quick-Boruvka Heuristic (based on MST algorithm by Boruvka)
 - .. Sort vertices in G arbitrarily
 - .. For each vertex v in G' with $\text{degree}(v) < 2$, add minimum weight edge e incident to v , if e in G but not in G'

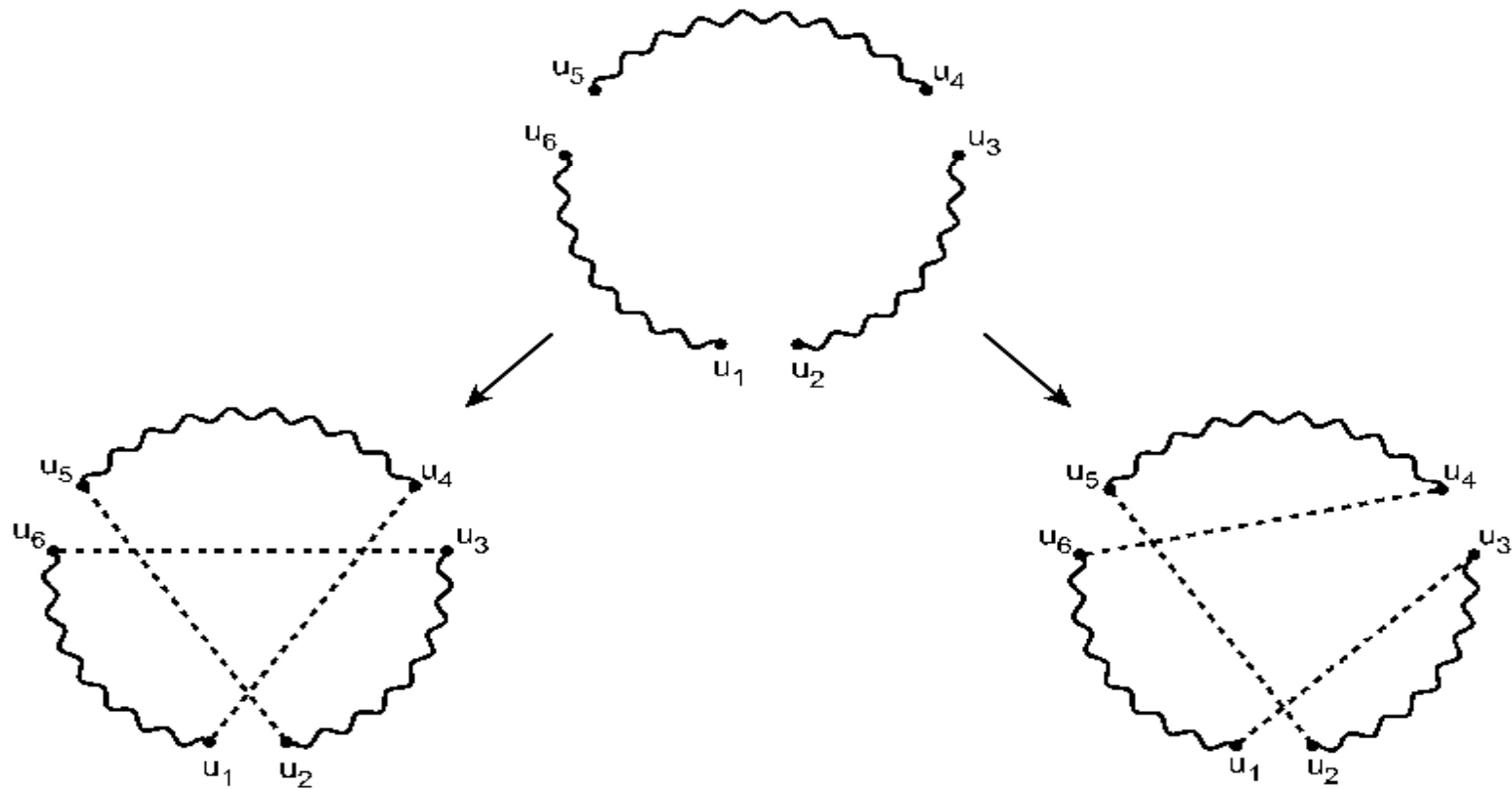
- n Savings Heuristic
 - .. Choose base vertex v_b and then make $n-1$ cyclical paths (v_b, v_i, v_b)
 - .. While there are cyclical paths left, combine two cyclical paths p_1 and p_2 by removing a common edge
 - .. Choose edge that results in maximum reduction of cost between new merged path p_{12} and the combined cost of p_1 and p_2



k-exchange Iterative Improvement

- n Candidate solution s and s' are neighbours iff s' can be obtained from s by deleting k edges
- n $k=2$ and $k=3$ are the most common choices
 - .. Also known as 2-opt and 3-opt
 - .. $k>3$ often returns better tours, but increased computation time renders the approach ineffective
- n At each step, examine all combinations of k edges
- n Tours that are locally 3-opt are also locally 2-opt

Typical 3-exchange move





Speed-up Techniques for 2 & 3-opt

n Fixed radius search

- .. In 2-opt, for a given vertex u_i , consider each of the two tour neighbours u_j
- .. Search around u_i for vertices u_k that are closer than $w(u_i, u_j)$
 - à Fixed Radius Near Neighbour Search
- .. Accept the first improving 2-exchange move
- .. Can extend to 3-opt

n Don't look bits (DLB)

- .. If no improving steps for a given vertex v was found, then until an incident edge is changes, don't consider v
- .. Use a DLB to indicate the vertex can be ignored
- .. After each search step, reset DLB's for all vertices incident to edges that were modified
- .. Significantly reduces time complexity of first-improvement search



Speed-up Techniques for 2 & 3-opt

n Candidate lists

- .. Efficient access to a list of neighbouring vertices for a given vertex, usually sorted by edge weight
- .. Makes Fixed-Radius search very efficient
- .. Often desirable to bound length of candidate list
- .. Simple k-lowest weight edges sometimes problematic
- .. Alternatives are quadrant nearest-neighbour list and Delaunay triangulations or a -value based lists

Effects of Speed-up Techniques

fr=fixed radius, dlb=don't look bits, cl=candidate lists

instance	2-opt-std		2-opt-fr+cl		2-opt-fr+cl+dlb		3-opt-fr+cl+dlb	
	Δ_{avg}	t_{avg}	Δ_{avg}	t_{avg}	Δ_{avg}	t_{avg}	Δ_{avg}	t_{avg}
kroA100	8.9	1.6	6.4	0.5	6.6	0.4	2.4	4.3
d198	5.7	6.4	4.2	1.2	4.3	0.8	1.4	30.1
lin318	10.6	22.1	7.5	2.1	7.9	1.5	3.4	65.5
pcb442	12.7	55.7	7.1	2.9	7.6	2.2	3.8	63.4
rat783	13.0	239.7	7.5	7.5	8.0	5.8	4.2	213.8
pr1002	12.8	419.5	8.4	13.2	9.2	9.7	4.6	357.6
pcb1173	14.5	603.1	8.5	16.7	9.3	12.4	5.2	372.3
d1291	16.8	770.3	10.1	16.9	11.1	12.4	5.5	377.6
f11577	13.6	1251.1	7.9	25.8	9.0	19.2	4.0	506.8
pr2392	15.0	2962.8	8.8	65.5	10.1	49.1	5.3	878.1



The Lin-Kernighan (LK) Algorithm

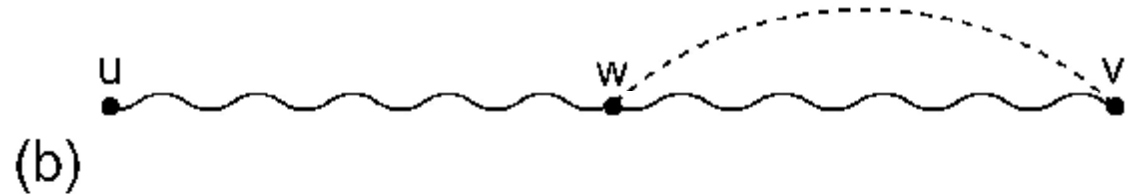
- n k-exchange neighbours with $k > 3$ sometimes gives better results, but at the cost of much higher computation time
- n LK constructs complex search steps by iteratively concatenating smaller elementary 1-exchange moves
- n In each complex step, a set of edges $X = \{x_1, \dots, x_r\}$ is deleted from current tour p , and replaced by another set of edges $Y = \{y_1, \dots, y_r\}$ to form a new tour p'
- n The number of edges that are exchanged, r , is variable and changes from one complex step to another

The Lin-Kernighan (LK) Algorithm

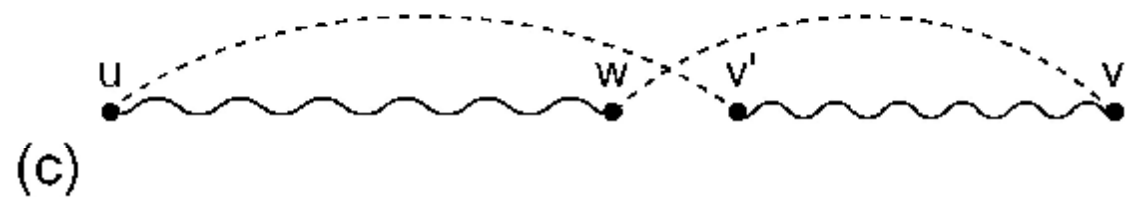
a) An edge (u,v) is removed to create a H -path



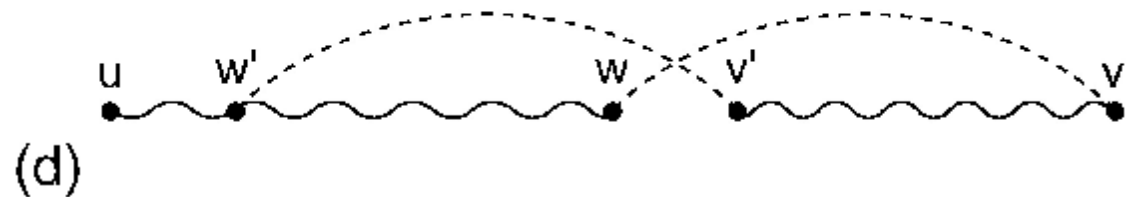
b) A new edge (w,v) is added to create a d -path



c) d -path broken by removing edge (w,v') , new H -cycle created by adding (u,v')



d) Or edge (w',v') could be added instead to create another d -path





The Lin-Kernighan (LK) Algorithm

n At each stage in the construction process, compute:

- .. $w(p_i)$ = length of tour p_i
- .. g_i = gain criteria, defined as:

$$g_i = \sum_{j=1}^i w(y_j) - w(x_j)$$

n Stop construction when $g_i < w(p) - w(p_{i^*})$, where p is current tour and p_{i^*} is best tour encountered during the construction



The Lin-Kernighan (LK) Algorithm

n Restrictions/constraints

- .. Sets X and Y are disjoint (bounds length of search)
- .. Limited backtracking: only for first two levels, i.e. x_1, y_1, x_2, y_2
- .. Ensures all 2- and 3-exchange moves are checked when searching for improving steps

n Pruning Techniques

- .. Search for edges (v, v') to be added to Y is restricted to the 5 shortest edges incident to v
- .. Basic look-ahead: choose edges to be added to Y such that $w(x_{i+1}) - w(y_i)$ is maximize



Variants of the LK Algorithm

- n Variations of algorithm differ mainly on:
 - .. Depth and width of backtracking
 - .. Rules used for look-ahead
 - .. Rules used to bound complex steps
 - .. Choice of 2-exchange move as elementary step

- n Variations do not always lead to optimal performance

- n Noteworthy variation is Heslgaun's LK (LK-H)
 - .. First-improvement on 5-exchange move
 - .. Candidate list based on a -values
 - .. Backtracking limited to one edge



Iterated Local Search for the TSP

- n Straight-forward flexible way of extending simple SLS methods
- n ILS and some hybrids, are among the best-performing TSP algorithms
- n Some examples of ILS algorithms:
 - .. Iterated Descent
 - .. Large-Step Markov Chains (LSMC)
 - .. Iterated Lin-Kernighan
 - .. Chained Lin-Kernighan
 - .. Iterated Helsgaun (ILK-H)



Large-Step Markov Chains (LSMC)

- n Developed by Martin, Otto and Felten, was the first high-performance ILS algorithm for the TSP
- n Sequence of local minimum can be modeled as a Markov chain
- n Segment of search trajectory between subsequent local minima corresponds to a “large step”
- n Uses a random double-bridge move as perturbation
 - .. Only considers move if combined weight of new edges is less than a constant k times average weight of optimal tour



Large-Step Markov Chains (LSMC)

- n Local search step was originally a 3-opt first improvement; later changed to a LK-variant with speed-up techniques
- n Acceptance criteria is same as Simulated Annealing
- n LSMC with 3-opt can solve small TSP instances with 200 cities in less than 1 CPU hour
- n LSMC with LK variant reduced time by a factor of 4 and found optimal solutions to several large TSPLIB instances
 - à found to critically rely on non-zero S.A. temperatures
- n A variant of LSMC that always accepted better tours was called Chained Local Optimization (CLO)



Iterated Lin-Kernighan (ILK)

- n An early variation on LSMC by Johnson and McGeoch
- n Several key differences:
 - .. Acceptance criteria always selects best of two candidate solutions
 - .. Double-bridge perturbation phase does not use edge-weight limitation; instead just choose 4 random cut points
 - .. Local search is initialized with a randomized greedy heuristic
 - .. Uses a much more efficient implementation of LK



Chained Lin-Kernighan (CLK-ABCC)

- n Like ILK, uses LK algorithm for subsidiary local search
- n Main differences from ILK:
 - .. Uses smaller candidate sets
 - .. Locally-restricted perturbation mechanism: geometric double-bridge move
 - .. Initializes the search using Quick-Boruvka construction heuristic
 - .. Resetting of the DLB's is applied to vertices within 10 edges of endpoints of modified edges; also include vertices in the neighbour sets (next slide)



Chained Lin-Kernighan (CLK-ABCC)

n The geometric double-bridge move works as follows:

1. Randomly sample a set of vertices from G : $U = \min\{0.001 * n, 10\}$
2. For each edge $(u, \text{succ}(u))$, where $u \in U$, remove the edge with the maximum difference $w(u, \text{succ}(u)) - w(u, u^*)$ where $u^* =$ nearest neighbour of u
3. Choose 3 vertices uniformly at random from the k nearest neighbours of u , and remove the edges $(u_i, \text{succ}(u_i))$ for $i=1,2,3$
4. The 4 edges chosen above determine the double-bridge move

à Value of k controls the locality of perturbation



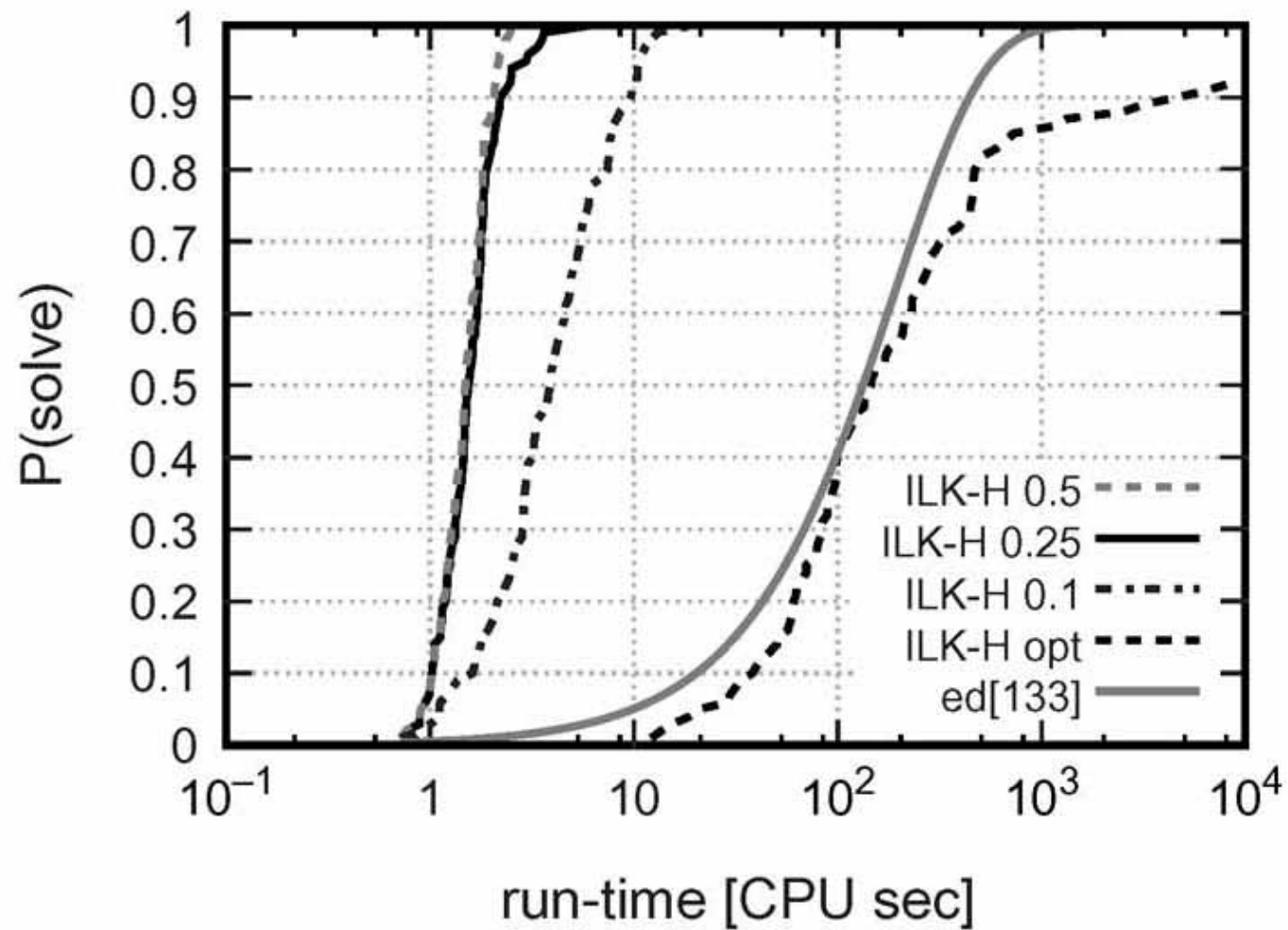
Iterated Helsgaun (ILK-H)

- n Uses LK-H for subsidiary local search procedure
- n One of the best-performing SLS algorithms for TSP in terms of the solution quality
- n Perturbation mechanism is based on a construction heuristic, biased by the incumbent candidate solution
- n Acceptance criterion only accepts tours that lead to an improvement over the incumbent candidate solution
- n Uses hashing technique to efficiently check if a given solution was previously found to be a local optimum

Perturbation Mechanism for ILK-H

```
procedure constructionILK-H( $G, \hat{s}$ )  
  input: weighted graph  $G := (V, E, w)$ , incumbent candidate solution  $\hat{s}$   
  output: candidate solution  $s \in S(\pi')$   
   $p :=$  empty tour;  
   $u_i :=$  selectVertexRandomly( $V$ );  
    append vertex  $u_i$  to partial tour  $p$ ;  
  while  $p$  is not a complete tour do  
     $C := \{u_j \mid (u_i, u_j) \text{ is a candidate edge} \wedge \alpha((u_i, u_j)) = 0 \wedge (u_i, u_j) \in \hat{s}\}$ ;  
    if  $C = \emptyset$  then  
       $C := \{u_j \mid (u_i, u_j) \text{ is a candidate edge}\}$ ;  
    end  
    if  $C = \emptyset$  then  
       $C := \{u_j \mid u_j \text{ not chosen yet}\}$ ;  
    end  
     $u_j :=$  choosePseudoRandomVertex( $C$ );  
    append vertex  $u_j$  to partial tour  $p$ ;  
     $u_i := u_j$ ;  
  end  
  return  $p$   
end constructionILK-H
```

Performance of ILK-H





Variations to ILS for TSP

- n Other perturbation mechanisms
 - .. Single random k-exchange steps
 - .. Modification of instance data
 - .. Genetic Transform (GT)

- n Other acceptance criteria
 - .. LSMC uses Metropolis acceptance criterion
 - .. Dynamic Restart Criterion
 - .. Fitness-distance diversification (fdd)

- n Tour Merging



Fitness-distance Diversification

- n Restarting search from a new initial candidate solution requires t_{init} time to reach high-quality solution
- n Instead, FDD achieves diversification by finding high-quality solutions beyond a minimum distance from the incumbent tour
- n Let $d(s,s')$ = the bond distance between tours s and s' , which returns the # of edges in s but not in s' (or vice-versa)



The FDD Function

1. Generate a set P comprising p copies of \hat{s}
2. Apply one perturbation step and a subsidiary local search to each candidate solution in P
3. Let Q be the set of q highest-quality candidate solutions from P , where $1 \leq q \leq p$
4. Let s' be the candidate solution from Q with maximal distance to \hat{s}
if $d(s', \hat{s}) < d_{\min}$ or if $\text{numIterations} < \text{maxIterations}$
 go to step 2
else
 return s'



Tour Merging

- n When randomly restarting the search process, all information from high-quality tours is lost
- n Tour Merging is a hybrid algorithm that tries to utilize information collect from multiple runs
- n Phase 1:
 - .. Generate a set of T of very high-quality tours
 - .. Form a new graph $G' = (E', w')$ where E' is edges contained in at least one tour in T
- n Phase 2:
 - .. Find an optimal tour through G'



Population-based SLS Algorithms

- n Iteratively manipulate populations of solutions
- n Tour Merging is an extreme example of this
- n Some common algorithms are:
 - .. Population-based ILS
 - .. Evolutionary Algorithms
 - .. Memetic Algorithms
 - .. ACO Algorithms



Population-based ILS Algorithms

- n Extend basic ILS by applying main search steps to each individual of the population (i.e. each candidate solution)
- n Uses selection and mutation, but not recombination
- n Interaction between populations is fairly limited
 - à allows for efficient parallel implementations



Population-based ILS Algorithms

- n Stützle examined three different algorithms with varying degrees of interaction
 1. No interaction: performs fixed number of independent ILS runs
 2. Replace-Worst: After l iterations, copy of best tour replaces worst tour
 3. ss-ILS: Performs j iterations of standard ILS to one selected tour s_0 from population. If improvement found, replace a tour in population with s_0



The Memetic Algorithm (MA-MF)

- n Memetic Algorithm by Merz and Freisleben
- n Original algorithm has been improved by more efficient local search procedures, better recombination operators, and the addition of diversification mechanisms

Construct initial population
Perform subsidiary local search
While terminate criterion not satisfied
 Recombination
 local search
 Mutation
 local search
 Selection
 local search



MA-MF : Initializing the Population

- n Initial population is constructed from a randomized variant of the Greedy Heuristic
- n For each tour in population, iteratively insert $n/4$ edges ($n = \#$ of vertices), where edges are chosen as follows:
 - Select uniformly at random a vertex v from G (not already in partial tour)
 - Shortest feasible edge to v is selected with probability $2/3$; otherwise second shortest feasible edge is selected
 - à Feasible edge: endpoints not already contained in partial tour
- n Complete the partial tour with standard GH



MA-MF : Recombination Procedure

- n Various recombination operators have been used, but Greedy Recombination (GX) achieves best performance
- n GX operator generates one offspring from two parents and consists of four phases:
 1. Copy a fraction p_e of edges common to both parents
 2. Add a fraction p_n of new short edges not contained in either of the parents
 3. Copy another fraction p_c of short edges from parents
 4. If necessary, complete candidate tour using randomized greedy heuristic



MA-MF : Mutation and Selection

- n Mutation operator is the standard double-bridge move, applied to a set of candidate solutions chosen uniformly at random

- n Uses a $(\mu + I)$ selection strategy:
 - .. μ = population size
 - .. I = number of new tours generation from recombination and mutation
 - .. Choose the best μ tours from $\mu + I$

- n Restart operator used conditionally to maintain diversity (a random k-exchange move with $k=0.1 * n$)
 - .. If average distance between tours falls below a given threshold
 - .. Average solution quality has not changed in the last 30 iterations



Results for MA-MF

- n Found optimal solutions for all TSPLIB instances up to 1000 vertices with average runtime of 2 CPU minutes
- n For all larger TSPLIB instances, solution-qualities within 1% of optimum were reached with average runtime of 1 CPU hour
- n For instances with more than 5,000 vertices, MA-MF does not perform as well as high-performance ILS algorithms
- n MA-MF could be more competitive if local search procedure was replaced with a more effective LK algorithm



ACO Algorithms for the TSP

- n TSP has been critical for the development of Ant Colony Optimization (ACO) algorithms
- n Some more well-known ones are:
 - .. Ant System
 - .. Ant Colony System
 - .. MAX-MIN Ant System
- n All used TSP as first example application
- n All successful ACO algorithms include:
 1. Good balance between search intensification and diversification
 2. Subsidiary local search on ant-constructed tours

MAX-MIN Ant System (MMAS)



- n Borrows construction step from Ant System (AS)
- n Three major differences from AS:
 1. Strongly exploits the best candidate solution during pheromone update → elitist strategy
 2. Upper and lower limits on pheromone trail levels
 3. Pheromone trail levels initialized to encourage exploration
- n Can solve instances from 100's of vertices to more than 1000 in minutes to hours, respectively.
- n Still, not competitive with state-of-the-art ILS algorithms



Summary

- n TSP is a central problem in combinatorial optimization
- n State-of-the-art complete TSP algorithms solve instances with several thousand vertices, while SLS algorithms solve instances with several millions within a fraction of a percent from optimum
- n Construction heuristics find reasonably good solutions extremely fast, iterative improvements methods and VDS method, such as Lin-Kernighan, serve as a basis to many other more advanced TSP algorithms
- n Iterated Local Search are the best-performing TSP algorithms
- n Population-based algorithms appear to be promising as the basis for more advanced TSP algorithms



References

- n “TSP Homepage”. <http://www.tsp.gatech.edu/>
- n Eric Baum. *What is Thought?*. MIT Press, 2004.
- n Holger H. Hoos and Thomas Stuetzle. *Stochastic Local Search Foundations and Applications*. Morgan Kaufmann, 2005.
- n Ant image borrowed from www.antnest.co.uk/Slytherin.html
- n Title graphic borrowed from www.ghchemicals.com