The Problem

- n points on a line
- these define n choose 2 distances
- given points, find distances: easy
- given distances, find points: not so easy
Example

Distances

11 10 9 8 7 6 6 5 5 4 3 2 2 1 1
The Outside-In Algorithm

- Largest distance gives leftmost and rightmost points.

- Next largest distance defines the next inside point; should we put it on the left or on the right?
  - 2 copies: both left and right;
  - 1 copy: either is fine by symmetry

- Next largest distance defines the next inside point; should we put it on the left or on the right?
  - 2 copies: both left and right;
  - 1 copy: either is NOT fine as symmetry is already broken
  - How do we choose the right one?
  - Maybe there is no genuine choice, each choice leading to a distinct solution
The Outside-In Algorithm

- **Choosing Left or Right**
  - Distances from new point to points in the SymBlock: doesn’t matter
    - Remove these distances from consideration
  
  - Distance $x$ to the symmetry breaker: matters
    - If $x$ does not exist then place on left
    - If $x$ exists then: can’t say, maybe place on left and then $x$ is realized by a later point along with a point in the symblock
The Outside-In Algorithm

Algorithm

- Two choices at each step, sometimes locally undisambiguatable
- Try both choices; take one first and then backtrack if you hit a dead end
- Time: $2^n$ backtracking alone with each step requiring a dead-end check

**dead end check:** check if distances from this point to all previous points are available, and then mark these distances as unavailable, requires $n \log n$ time

**Total Time:** $O(2^n n \log n)$
Status of the Problem

- Backtracking $2^n \times \log n$ is best known provable worst case solution
- Open Problem: Can one do better?
Suppose the $2n-3$ distances which come from endpoints are identified

- Easy Exercise??
Special Cases

- Suppose all distances are distinct?
  - Not a Hard Exercise??
Zhang’s Example: Backtracking is not of limited depth
All distances between action blocks are available between the future anchor blocks and the sym blocks
Special Cases

- Zhang’s Example: Solvable in polynomial time!!!
hint: do 2 passes

  - first pass: for each locus, identify whether or not it is symmetric
  - second pass: do the actual placement

How about these extended Zhang examples
Why doesn’t the 2 pass work here?
Open? Could be one way to approach the problem.
A Novel Approach

Polynomial Representation

- $d_1, \ldots, d_m$ are the distances
- $p_1, \ldots, p_n$ are the point locations
- $P(x) = \sum_k x^k p_k$
- Then
- $P(x)P(1/x) = \sum_k x^k p_k \sum_k x^{-k} p_k$
  $= \sum_{i,j} x^{p_i - p_j}$
  $= \left[ \sum_m (x^{d_m} + x^{-d_m}) \right] + n$, this is known, call it $D(x)$

So turnpike comes down to factoring $D(x)$ into factors $P(x)$ and $P(1/x)$ with integer or even 0/1 coefficients.
Turnpike via Polynomial Factorization

- Integer Polynomial Factorization runs in time polynomial in the degree
- Degree of $D(x)$ is the largest distance
- So if the degree of $D(x)$ is small, say poly in $n$, then we have a polynomial time algorithm for the turpike problem.
- What happens if the degree of $D(x)$ is large? Degree could be exponential in $n$ or even super exponential (i.e., $2^{\{n^2\}}$) while keeping the problem size polynomial in $n$.
- So how do we factor high degree polynomials in time sub-exponential in the number of monomials rather than the degree??
Bounding Degrees

- Transform the given set of distances to a set of smaller distances so that the solution set does not change. Denote this transformation by $T()$

- Key Observation: Suffices to ensure that $d_i + d_j = d_k$ implies $T(d_i) + T(d_j) = T(d_k)$ for all triples of distances $d_i, d_j, d_k$

Proof: Exercise??

- Now one solves the above set of linear equations, numbers in the solution have size at most $6^{(n-1)/2}$

Proof: Exercise?? Hint, rewrite all equations in terms of just $n-1$ distances, i.e., those from the left endpoint; so this system has rank only $n-1$ and up to 6 terms in each equation; use cramer’s rule and hadamard’s ineq.
How Many Solutions?

How many ways can an integer polynomial $D(x)$ be factorized over integers into the form $P(x)P(1/x)$?

- Note $D(x)$ is reversible, i.e., $D(x) = \text{deg}(D(x)) \cdot D(1/x)$
- $D(x)$ is uniquely factorizable into irreducible factors over the integers. Proof: Exercise??
- Each irreducible factor is either reversible or irreversible.
- If $P_i(x)$ is an irreversible factor, then $P_i(1/x)$ is also an irreversible factor
- Reversible factors must repeat an even number of times (provided there is a solution).
- The total number of distinct solutions is $2^{\text{number of irreversible factors}}$.
- The number of irreversible factors is $O(\log n)$
- So the number of distinct solutions is just polynomial in $n$!!!
Number of Irreversible Factors?

- Define a certain measure of a polynomial

- Show the measure of $D(x)$ is small, i.e., bounded by not the degree but the number of terms, or the sum of absolute term coefficients, or sum of squares of these term coefficients.

- Show that the measure is multiplicative, so measure of product is product of measure when you multiply polynomials.

- Show that an irreversible polynomial has measure at least $1+x$ for some fixed value $x$.

- This implies the bound of $O(\log n)$ irreversible factors.
The number of solutions to a turnpike instance is polynomial in \( n \), which is roughly the input size.

Such problems belong to the complexity class FewP which is likely to be a strict subset of NP. So Turnpike is unlikely to be NP-Hard.

Does it give a subexponential time algorithm? Not yet.

We have an algebraic number theoretic approach which yields a subexp algorithm, assuming a certain number theoretic fact. An overview in later classes.
Skiena, Smith, Lemke: 95, show the bound on the number of solutions, must read paper, available on the web

Special cases: Distances from endpoints identified
Pandurangan, Ramesh: on my website

Unique distances and the alg num th approach
Mangesh, Naidu, Ramesh: being written up, also in Mangesh’s thesis
Thank You