Introduction to Optimization

Cynthia Phillips
Sandia National Laboratories
caphill@sandia.gov
March 15, 2011

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

(Combinatorial) Optimization

Find the best solution out of a (finite) set of feasible solutions.

Begins with an English description

Solution: What decisions can you make? What decisions are implied by the actual decisions?

Feasible: What constraints must a solution satisfy?

Best: How do we compare two solutions? Is there a score?

Formal Modeling

- Variables
  - Decision variables
  - Helper variables
- Objective Function: How good is a solution?
  - Multiple objectives
- Constraints
  - Requirements for feasibility
  - Can include goals
- Input parameters
  - Data, for evaluating a solution, determining feasibility

- Can take considerable effort
- Tradeoff: model fidelity vs tractability

Modeling

- Solution difficulty hierarchy
  - Black box
  - Has derivatives
  - Nonlinear constraints and/or objective
  - Convexity
  - Linear constraints and objective
    - Can have integer variables (MILP)
    - All continuous variables (LP)
  - Specific tractable problems: network flow, matching, matroids

- As structure becomes more restricted
  - Faster
  - Closer to optimal
- Even within the same “class” formulation matters

“Solving” an Optimization Problem

Solution strategy and measure of “success” depend on

- How fast the computation must run
  - Platform
- Development time
- Special structure of data
- Required degree of optimality
  - feasible? better? (near) optimal?
  - How important is each run?
  - Irrevocable? Recourse?
  - How good is the data?
  - How good is the model?

Confidence in Solutions

- Proof of (near) optimality
  - Mathematical
  - Computational
- Benchmarks
  - Verification: Am I solving the problem right?
  - Validation: Am I solving the right problem?
  - What is ground truth?
    - Simulation
    - Experiments
- Do not optimize past the confidence in the model and data

Multiple Solutions

- Exploring the space of near optimal solutions
  - When objective pressure is heuristic or stochastic
    - Evolution
  - Unexpressed goals
    - Not easily express mathematically
    - Maybe don't know yet
- Diversity
- Robust/stable point

Computation for insight

Options When a Problem is Hard

For intelligent search (almost all solvers have in some form):

- Search harder
  - Parallelization
- Search smarter
  - Understand/recognize/use (sub)-structure
  - Customize the solver
- Lower expectations
  - Approximations

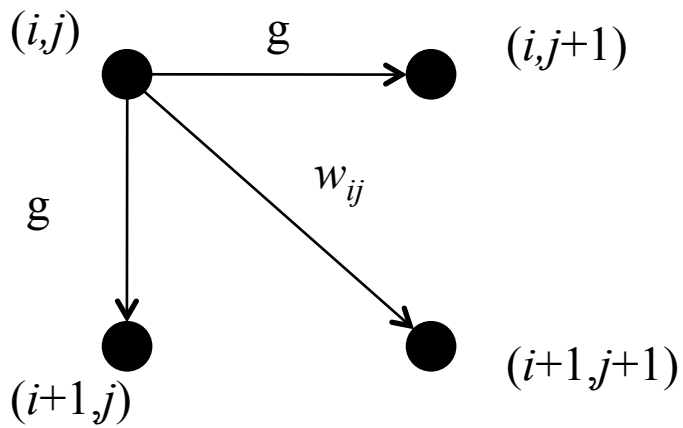
Example 1: Sequence Alignment (Naor, Brutlag)

- Edit distance score
  - Match score (w_a), positive
  - mismatch score (w_{ab}), negative (depends on similarity)
  - Gap score, negative
- Evolutionary justification, but not "correct"

--	A	C	C	T	G	C
G	A	--	C	A	G	--
g	w_A	g	w_C	w_{TA}	w_G	g

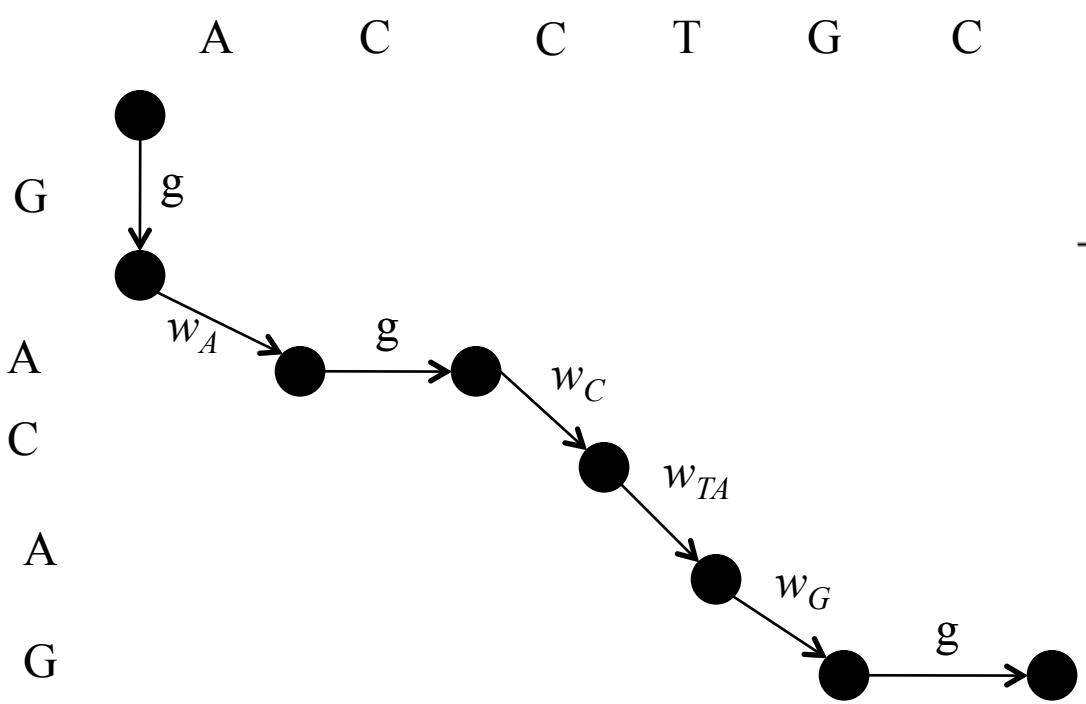
Graph Representation

- Node for each pair (i,j): ith element first sequence, jth from second
- Each node has 3 outgoing edges
  - Diagonal for a (mis)match
  - Horizontal/vertical for a gap insertion



Graph Representation

- An alignment is now a path from *s* to *t* in a 2D grid
- Goal: maximize path in a directed acyclic graph (tractable)
- Exponential number of s-t paths, though not all are high-score



--	A	C	C	T	G	C
G	A	--	C	A	G	--
g	w_A	g	w_C	w_{TA}	w_G	g

Compute Efficiently

- Dynamic Programming
- *D(i,j)* is score of best alignment through *i* elements of first string and *j* elements of second.

$$D(0,0) = 0$$

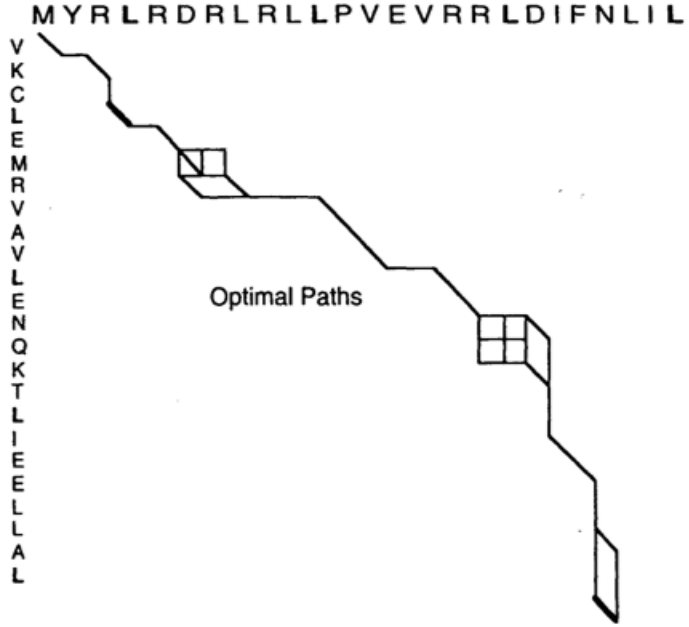
$$D(i,j) = \max\left(D(i-1,j) + g,\ D\left(i-1,j-1\right) + w_{ij},\ D\left(i,j-1\right) + g\right)$$

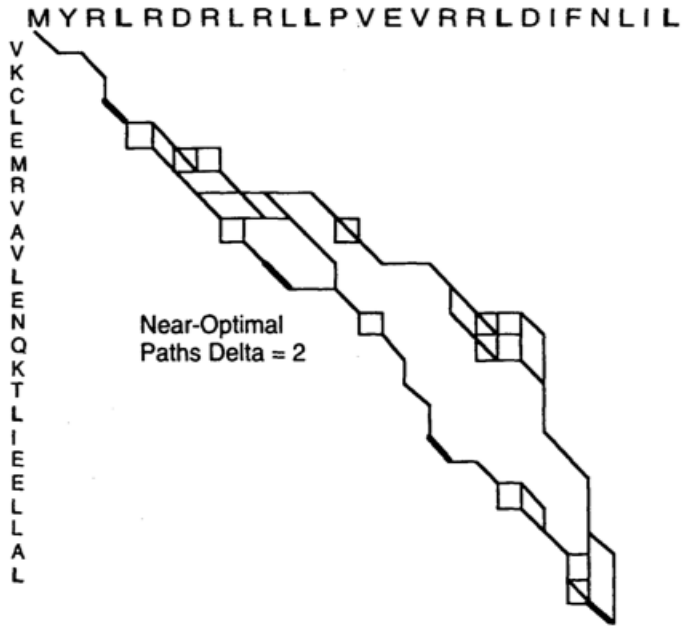
- Compute each element when ancestors done
- Can compute near optimal by doing the same in reverse *t* to *s*.

Compact Representation of near-optimal paths

- Include only edges on at least 1 path with score within Δ of optimal
- Δ>0 necessary to capture biology in general

Example: 2 Leucine zippers:

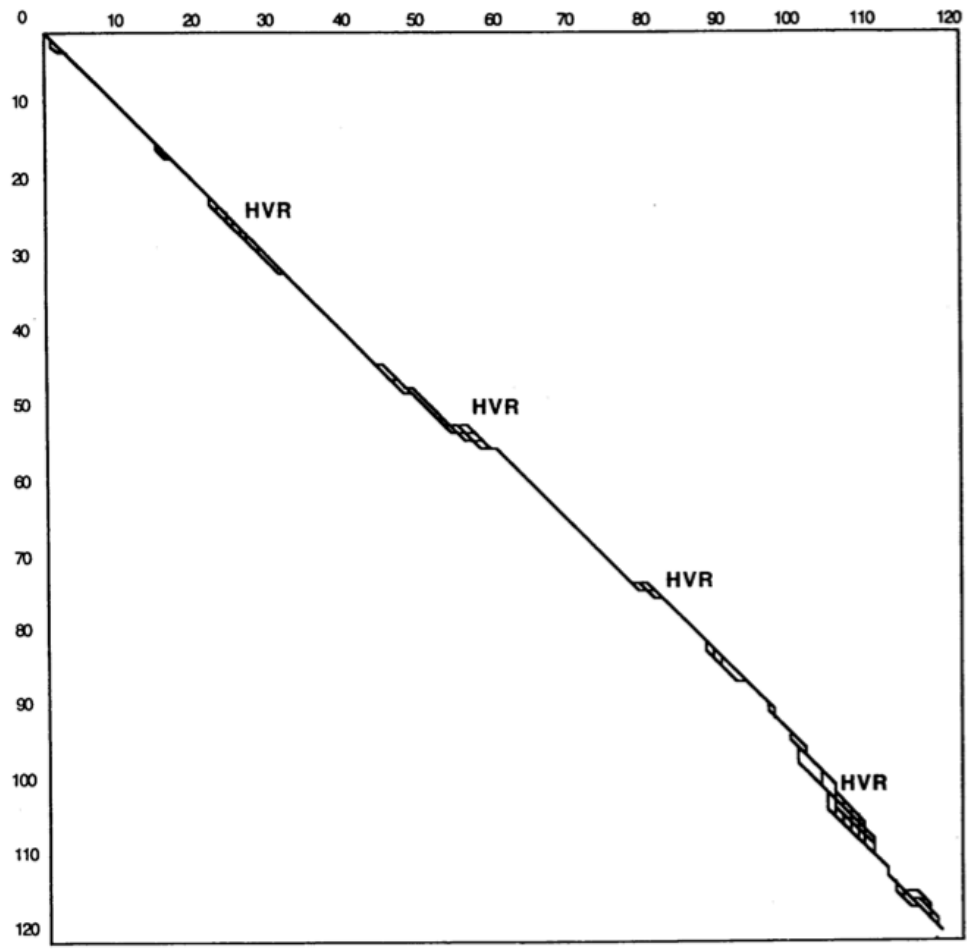




Sandia National Laboratories

Compact Representation

- Shows regions conserved in all near-optimal alignments.
- Example: heavy and light chain of human immunoglobulin (23% match, but similar structure when folded): Δ=2



Sandia National Laboratories

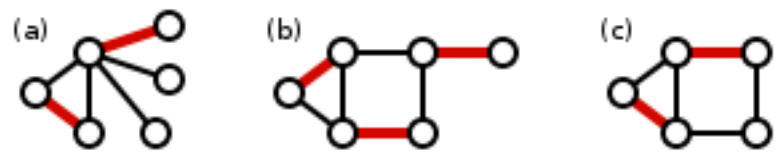
Other Nice Analytical Features

- Enumerate paths in score order (with weight transformation)
- Count number of near-optimal alignments
- Set of canonical paths
  - Polynomial number (mn, usually much less) for length-*n* and length-*m* strings
  - Covers the compact representation

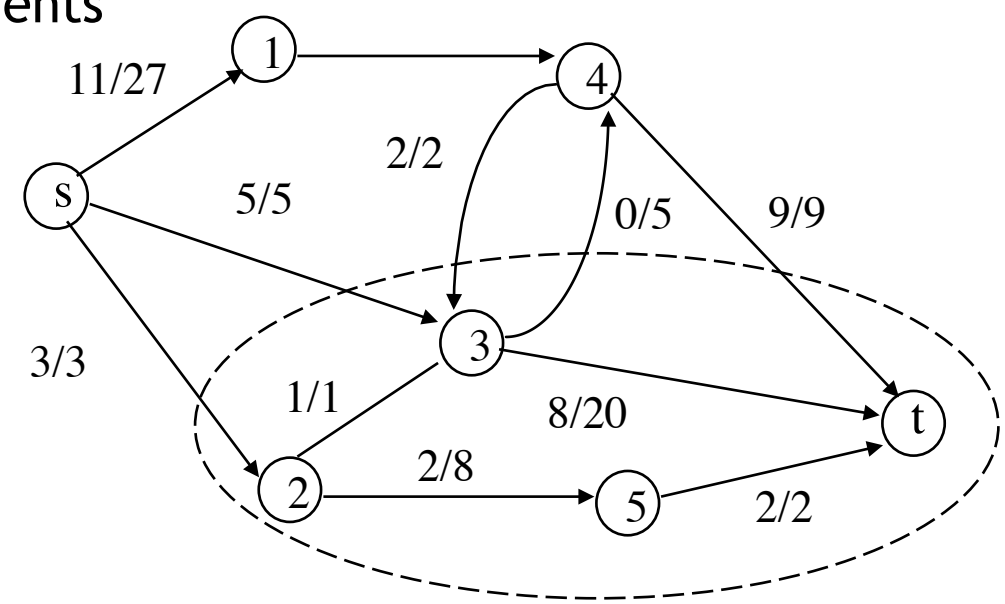
Graph Algorithms

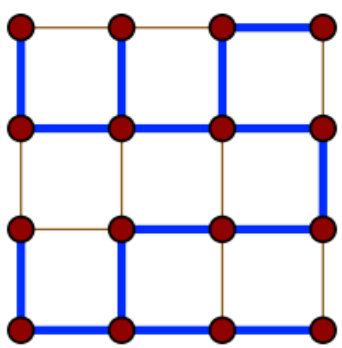
All these graph algorithms have efficient (polynomial-time) solutions

- Shortest paths
- Minimum cut/Maximum flow
- Minimum-weight spanning tree
- Random spanning tree
- (Strongly) connected components
- Biconnected components
- Planarity testing
- Matching
- Euler Circuit



(wikipedia)

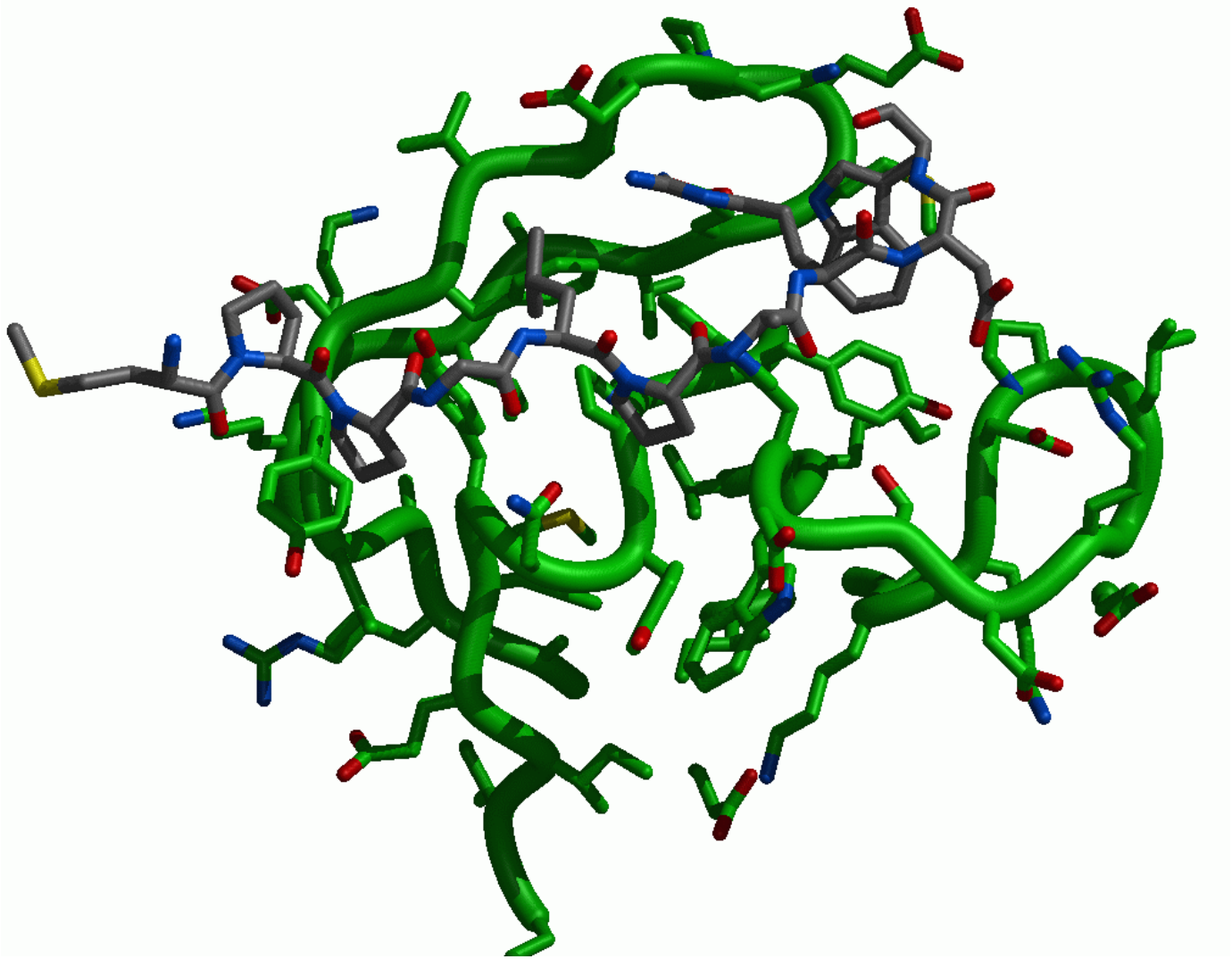




(wikipedia)

Sandia National Laboratories

Example 2: Protein-Peptide Docking (Hart,Roe)

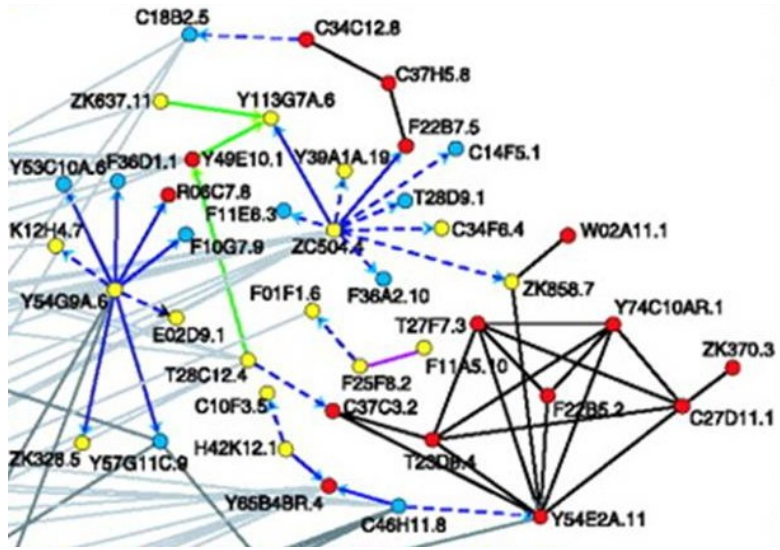


Motivation

Protein-protein interactions are essential to virtually all cellular processes

- Proteins are macro-molecules composed of a linear chain of amino acid residues
- Proteins folds into well-defined 3D structures that determine their role in cellular processes

Protein-protein interactions provide insight into protein function



Sandia National Laboratories

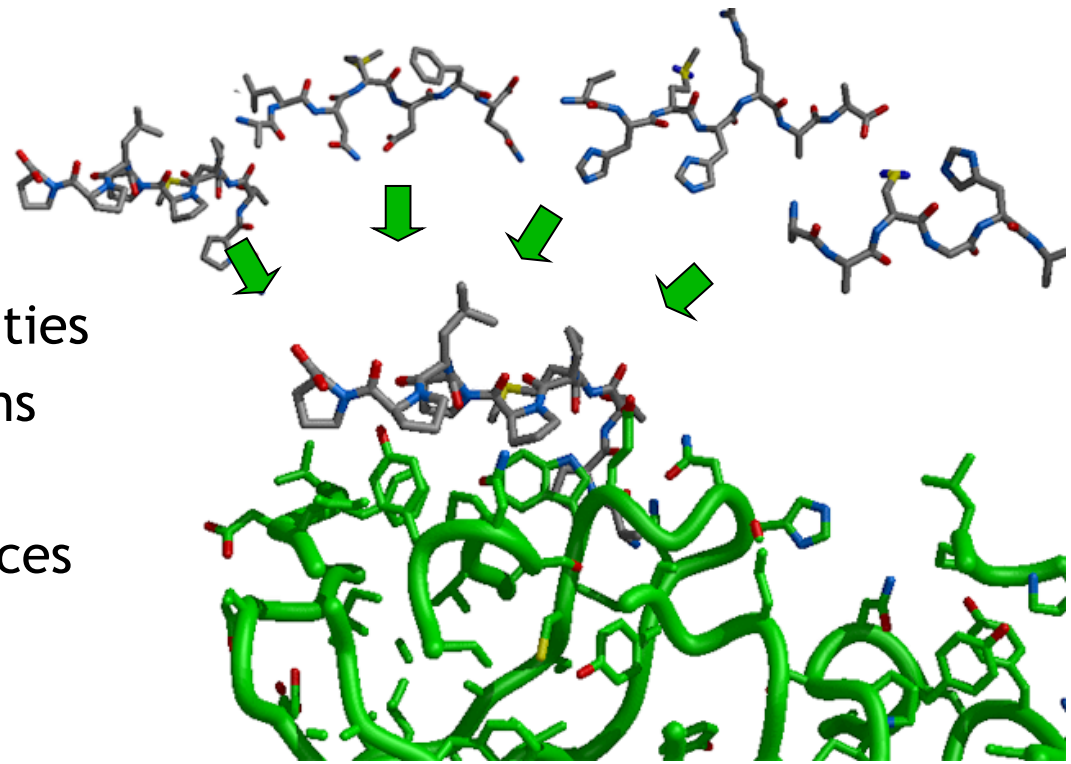
Phage-Display

Phage-display libraries: an experimental technique used to probe protein-protein binding interactions

- Libraries are typically of size 4-12 residues in length
- Libraries generate a consensus binding sequence

Goals:

- Describe potential binding partners
- Find the sequence specificities of these binding interactions



Impact: With binding sequences can search the genome for binding partners

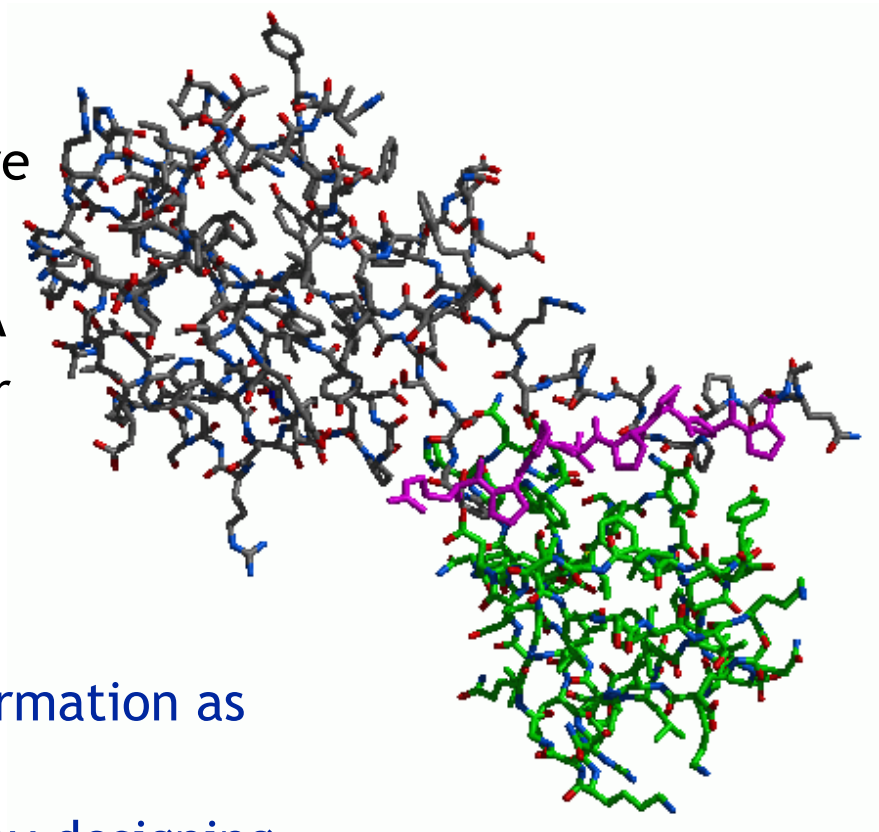
Computational Phage Display

Idea: study protein-protein interactions through protein-peptide docking

Many protein-protein interactions are mediated by modular domains

- PDZ, SH3, SH2, WW, PTB, FHA

These domains often bind to a linear stretch of the binding partner



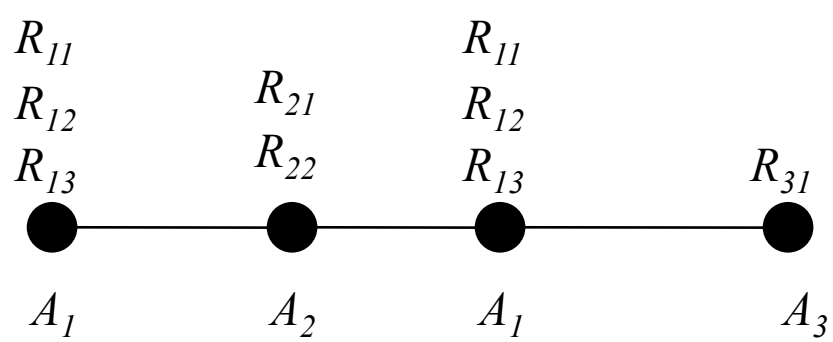
Impact:

- Provide the same type of information as experimental methods
- Minimize experimental costs by designing reduced "focused" phage-display libraries

Assumptions

1. Side-chain structures can be well-captured by rotamer libraries
   - Rotamer libraries taken from trusted data sources (e.g. PDB)
   - Discretize the structure prediction

2. The peptide backbone is well-constrained
   - We assume that it's fixed
   - If there is flexibility, then we could consider a (small) set of alternative backbone conformations



Peptide Structure Prediction

Goal: structure prediction for a given peptide

Variables: rotamer choices:

$$\delta_{ir} = \begin{cases} 1 & \text{if rotamer } r \text{ is assigned to sidechain } i \\ 0 & \text{otherwise} \end{cases}$$

Peptide structures are evaluated with an empirical energy model:

- Amber scoring function
- Generalized Born continuum solvation calculation

- Energy parameters:
  - E_{ir} = energy from having rotamer *r* in sidechain *i*
  - A_{irjs} = interaction energy from rotamer *r* in sidechain *i* and rotamter *j* in sidechain *s*

Mixed Integer programming (IP)

Min $c^T x$

Subject to: $Ax \le b$

$\ell \le x \le u$

$x = (x_I, x_C)$

$x_I \in Z^n$ (integer values)

$x_C \in Q^{n'}$ (rational values)

- Can also have inequalities in either direction (slack variables):

$$a_i^T x \le b_i \Rightarrow a_i^T x + s_i = b_i \ , \ \ s_i \ge 0$$

- Integer variables represent decisions (1 = yes, 0 = no)
- Surprisingly expressive
- Many good commercial and free IP solvers

Naturally an Integer Quadratic Program (IQP)

$$\min \quad \sum_{i,r} E_{ir}\delta_{ir} + \sum_{i,r,j,s} A_{irjs}\delta_{ir}\delta_{js}$$

$$\sum_{r} \delta_{ir} = 1$$

$$\delta_{ir} \in \{0,1\}$$

Note: many of the energies are negative, so this IQP is not convex

Note: this is a quadratic semi-assignment problem

An Integer Programming Formulation

$$\min \sum_{i,r} E_{ir}\delta_{ir} + \sum_{i,r,j,s} A_{irjs} w_{irjs}$$

$$s.t. \quad \sum_{r} \delta_{ir} = 1 \qquad \forall \text{ i}$$

$$\sum_{s} w_{irjs} = \delta_{ir} \qquad \forall \text{ i,r,j}$$

$$\sum_{r} w_{irjs} = \delta_{js} \qquad \forall \text{ i,j,s}$$

Note: this model can be refined to exploit data sparsity

Note: similar models have been derived by several other authors…

Finding Consensus Sequences

Problem:

- Empirical energies provide only a rough estimate of rotamer-rotamer or rotamer-protein interactions
- The discretization imposed by a rotamer library can create artificial infeasible interactions

Enumerate solutions:

- Within a certain percentage of the optimal objective value, or
- Better than some fixed cutoff value, or
- Among the n best solutions (ties broken arbitrarily)

Compute consensus matrix of amino acid frequency at each position

Using Consensus Information

I. Limit the scope of phage-display experiments
   - Only include amino acids at sites where they appear in near optimal solutions

II. Identify peptide docking candidates
   - Use a consensus matrix to score peptide sequences with an expected frequency
   - Can be use to scan the genome for binding partners
   - A similar approach has been taken using Boltzman energies to predict binding affinities
   - We expect that consensus information will prove more stable and predictive

Peptide Design

Same basic formulation

- The rotamer library at each site can include rotamers for all amino acids
- Optimizer implicitly selects an amino acid when selecting a rotamer
- ILPs for peptide design problems are much more difficult
  - Forrester and Greenberg describe better MILP models
- Teaser for Friday: Create a custom solver based on ILP search

Some Thoughts at This Point

The "answer" provided by an optimization solver is *not* simply the optimal solution!

Analysis of optimization results

"The purpose of computing is insight, not numbers."

R. W. Hamming

Examples

Is this solution relevant for a practical application?

- What is the fidelity of the computational model near this point?
- How sensitive is the model to perturbations?
- Do my input data accurately reflect real-world scenarios?

Why is this the optimal solution?

- Is this a global optimum?
- How distinct is the global solution?
- Why is this solution different?
- How do other solutions compare with respect to other design criteria?
- What is the global structure of the objective and constraints?

Challenges

Can we couple optimization with informatics strategies to provide insight into applications?

- How should we archive data generated during optimization?
  - branching decisions, local minima found, etc.
- What type of data analysis or visualization strategies can be used to interrogate these data sets?

How do we tailor optimizers to facilitate post-solution analysis?

- Is this more than simply printing more optimization data?
- How do we manage expensive data analysis computations?

Can we objectively critique this type of optimization research?

- We need more than 'horse race' comparisons
- How do we quantify 'insight'?

Decisions Given Uncertain Future

- Sometimes can express uncertainty with variable ranges or distributions
- Scenarios
  - Sample of possible futures
  - General technique when uncertainty is complex
    - Simulation-based
    - Truly stochastic
      - Weather
      - Congressional budgets at presidential discretization
  - Can improve answer as sampling improves

The Sensor Placement Problem

Issue: Contamination released in a municipal water network

Goal: develop early warning system

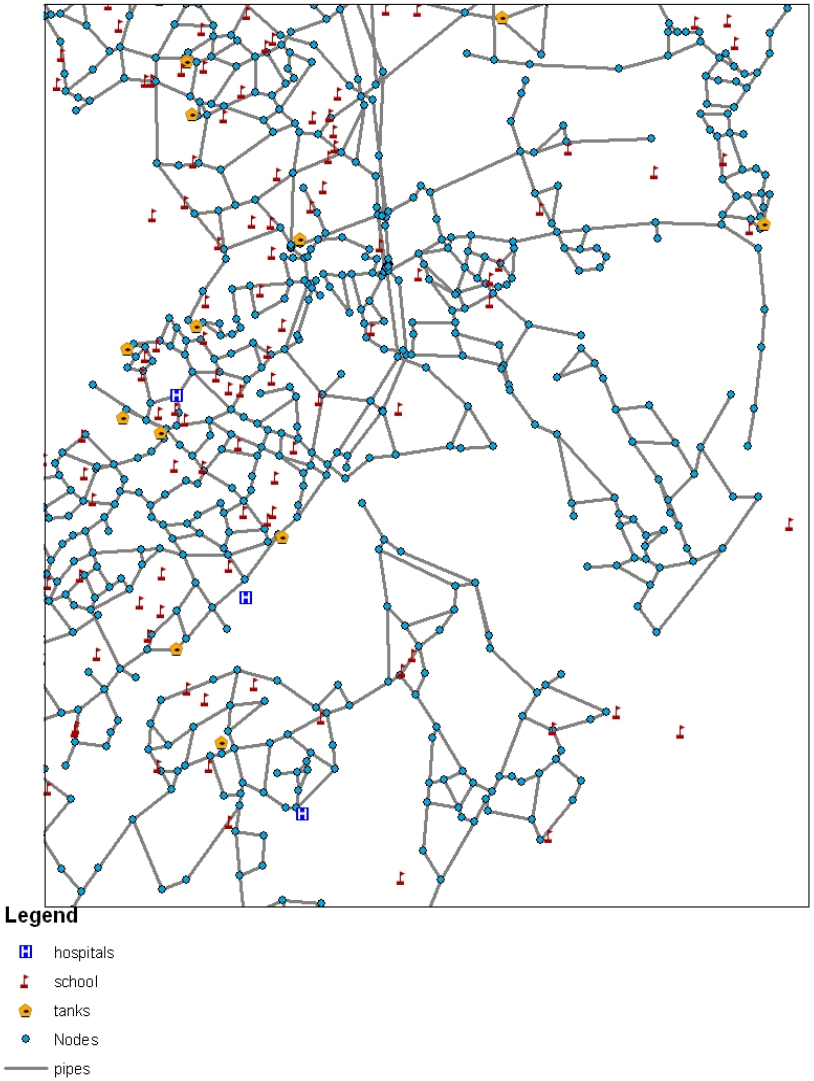
- Protect human populations
- Limit network remediation costs

Place sensors on

- Utility-owned infrastructure
- Schools
- hospitals

• Sensors are expensive

- Cost of sensors
- Cost of installation



Sandia National Laboratories

Contaminant Transport Modeling

Water movement (direction, velocity in each pipe) determined by

- Demand (consumption)
- Pumps
- Gravity
- Valves
- Sources/tanks

Current (most trusted) simulator

- EPANET code computes hydraulic equations to determine flows
- Discrete-event simulation for contaminant movement

Sensor Placement Modeling

- Data uncertainty
  - Aleatory uncertainty (inherent, uncontrollable)
    - Demand (drives water movement)
    - Population distribution
  - Epistemic (lack of knowledge)
    - Damage(costs, morbidity statistics)
    - Simulator fidelity
  - Both
    - sensor performance
    - attack distribution
      - Nature of contamination
        - When? Where? What? How much?

Modeling Assumptions

- Sensors are perfect
- Sensors raise a general alarm
  - Can model a response delay

- Fixed set of demand patterns for “typical” day
  - Seasonal variations
  - Special events
  - Weekday/weekend

Modeling Events

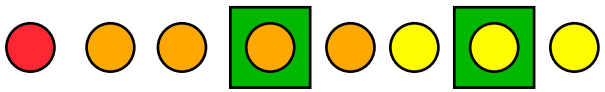
- Given: Set of events = (location, time) pairs
- Simulate the evolution of a contaminant plume
- For each event determine
  - Where/when event can be observed
  - Amount of damage prior to that observation

- Measures of damage/impact:
  - Population exposed
  - # deaths
  - Volume of contaminant release
  - Total pipe length contaminated
  - Time to detection
  - # failed detections

Witnessing an Event

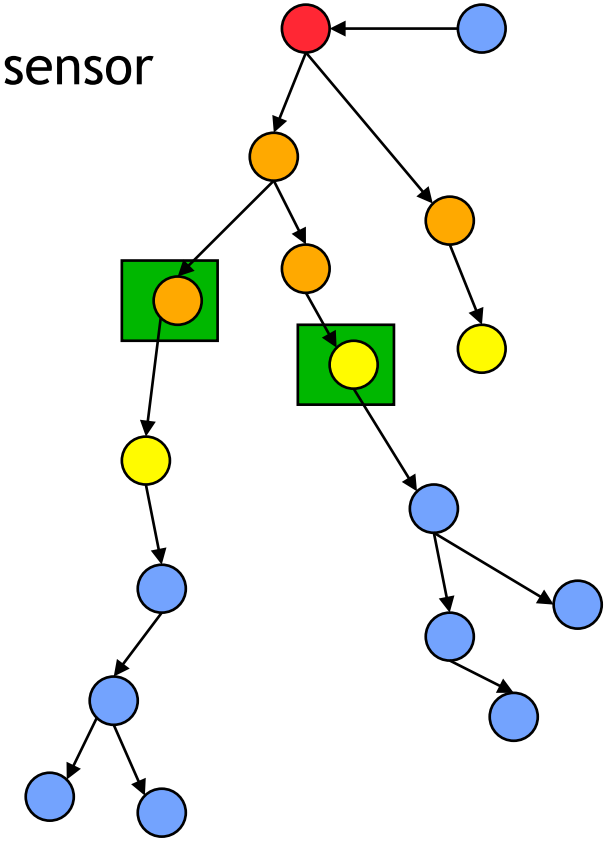
Simulator gives ordered list of nodes where a sensor could witness contamination

Witnesses:



This example has two (green) sensors.

Perfect sensor model: first sensor in list detects the event.



Evaluating a Sensor Placement

- Impact in red

d = dummy node (represents failure to detect)

	10	50	100	300	800
Event 1:	1	2	3	4	d
	10	150	400		1500
Event 2:	5	6	2		d
	10	10			200
Event 3:	4	7			d

Evaluating a Sensor Placement

- Impact in red

d = dummy node (represents failure to detect)

						Impact:
Event 1:	10	50	100	300	800	50
	1	2	3	4	d	
Event 2:	10	150	400		1500	400
	5	6	2		d	
Event 3:	10	10			200	200
	4	7			d	

Choose sensors 2 and 3 (black)

Mixed Integer programming (IP)

Min $c^T x$

Subject to: $Ax \le b$

$\ell \le x \le u$

$x = (x_I, x_C)$

$x_I \in Z^n$ (integer values)

$x_C \in Q^{n'}$ (rational values)

- Can also have inequalities in either direction (slack variables):

$$a_i^T x \le b_i \Rightarrow a_i^T x + s_i = b_i \ , \ \ s_i \ge 0$$

- Integer variables represent decisions (1 = yes, 0 = no)

One Sensor Placement IP for Water Networks

Variables:

$$y_i = \begin{cases} 1 & \text{if we place a sensor at location } i \in \mathcal{L}, \\ 0 & \text{Otherwise} \end{cases}$$

$$x_{ij} = \begin{cases} 1 & \text{if location } i \text{ raises the alarm (witnesses) event } j \\ 0 & \text{Otherwise} \end{cases}$$

Extreme points will have integer values for x_{ij} if the y_i are integral.

Each event has a dummy location to mark failure to detect

Objective function

Compromise across all "likely" event scenarios to minimize expected damage.

$$\text{minimize} \sum_{j \in A} \sum_{i \in L} \alpha_j w_{ij} x_{ij}$$

α_j – the weight of event $j = (i, t)$

w_{ij} – the total damage from event j if detected at location $i \in \mathcal{L}_j$

x_{ij} – 1 if location i raises alarm (witnesses) event j, 0 otherwise.

Sensor Placement Mixed Integer Program

$$\text{minimize} \sum_{j \in A} \sum_{i \in L_j} \alpha_j w_{ij} x_{ij}$$

$$s.t.$$

$$\sum_{i \in L_j} x_{ij} = 1 \qquad \forall j \in A \qquad \text{(every event witnessed)}$$

$$x_{ij} \le y_i \qquad \forall j \in A, i \in \mathcal{L}_j \qquad \text{(need sensor to witness)}$$

$$\sum_{i \in L} y_i \le p \qquad \text{(sensor count limit)}$$

$$y_i \in \{0,1\}$$

$$0 \le x_{ij} \le 1$$

Sensor Placement = p-median

p-median problem:

- n possible facility locations
- m customers
- d_{ij} = distance from customer j to location i

- Pick p locations and assign each customer to an open location to minimize the total distance.

Sensor placement as a p-median problem:

- Sensors = Facilities
- Network locations = potential facility locations
- Events = Customers to be "served" (witnessed)
- "Distance" from an event j to a node i = impact if a sensor at node i witnesses event j.

Formulation is really important in practice

In Unconstrained facility location (pick facilities to build and serve customers)

Part of formulation 1 is $y_{ij} \le x_i \quad \forall i, j$

Formulation 2 is the same except we sum these constraints over *i*:

$$\sum_{i=1}^{n} y_{ij} \le n x_j \quad \forall j$$

IPs are equivalent at optimality

But, (from Linderoth), for 40 customers, 40 facilities, random costs

- First formulation solves in 2 seconds
- Second formulation solves in 53,121 seconds (14.75 hours)

- Adding redundant constraints (even a lot) can be very helpful computationally

Model/Simulator Interaction

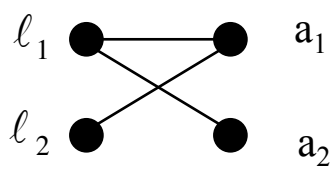
- Model requires only a list of witnesses and impacts for each event
- Model is stable as simulator improves
- EPANET has some known issues
  - Perfect mixing assumption
  - Numerical issues/scaling

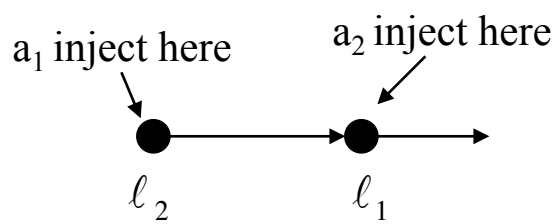
- Same basic model works in other settings
  - Airborne contaminants
  - Blog watching

The p-median Problem

- Open *p* facilities and assign each customer to an open facility to minimize the total customer->facility distance.

- NP-complete
- Well Studied
  - Operations Research heuristics
  - Approximation algorithms for metric p-median
- Water problem not metric
  - Doesn't satisfy triangle inequality
    - For bipartite graphs: weight of edge at most weight of path between endpoints





Imperfect Sensors

- Sensor a location *i* detects with fixed probability p_i
  - Assume independence (well spaced geographically)
- In practice, base on water quality zones
- False positives important
  - For this formulation handle by tuning (offline)

1	2	3	4	d	
10	50	100	300	800	Impact
.1	.3	.25	.5	1	Raw success probability p_i
.1	.27	.16	.24	.23	Witness probability if All 4 locations have sensors

- Witness an event if all sensors that see it first fail, and you succeed

Imperfect Sensors formulation (non-linear)

$$\text{(impSP) minimize} \sum_{a \in \mathcal{A}} \alpha_a \sum_{i \in \mathcal{L}_a} d_{ai} x_{ai}$$

$$\text{where} \begin{cases} \sum_{i \in \mathcal{L}_a} x_{ai} = 1 & \forall a \in \mathcal{A} \\ x_{aj} = p_j s_j \prod_{i \in \mathcal{L}_{aj}} (1 - p_i s_i) & \forall a \in \mathcal{A},\ j \in \mathcal{L}_a - \{q\} \\ \sum_{i \in L} s_i \leqslant p & \\ s_i \in \{0,1\} & \forall i \in L \\ 0 \leqslant x_{ai} \leqslant 1 & \forall a \in \mathcal{A},\ i \in \mathcal{L}_a \end{cases}$$

- x_{ai} = probability location *i* witnesses event *a*
- s_i = 1 if put sensor on location *i*
- d_{ai} = impact if location i witnesses event *a*
- p_i = success probability for a sensor at location *i*

One-Imperfect Witness Approximation

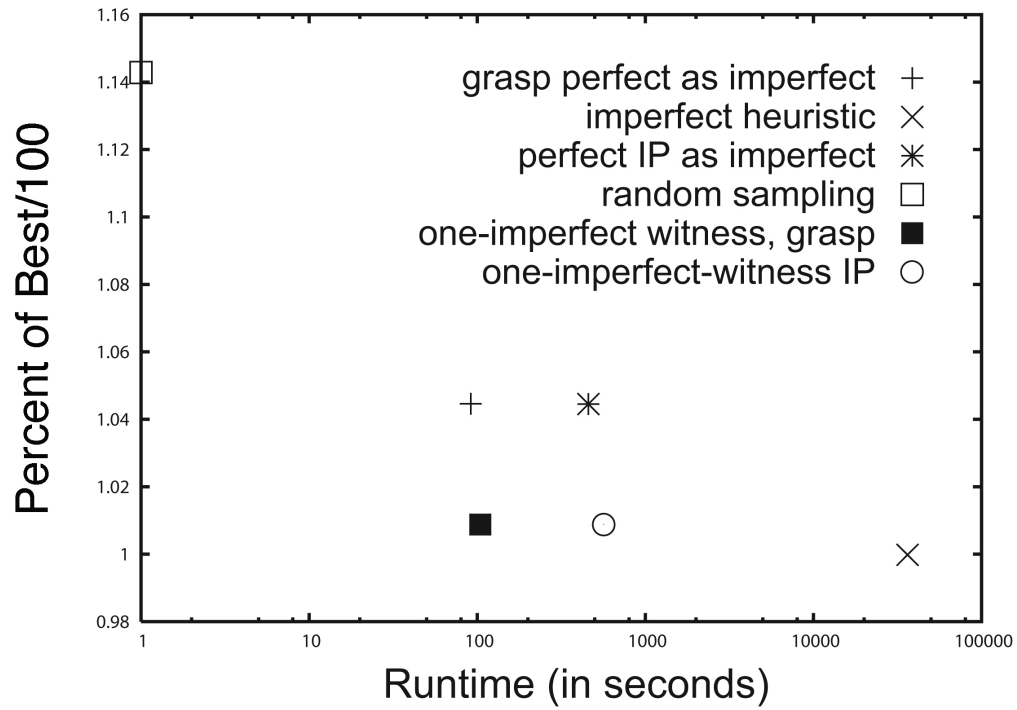
- Sensor a location *i* detects with fixed probability p_i
- Only consider the best sensor for each event
  - No "back up"
- Adjusted impact: $d'_{ai} \rightarrow p_i d_{ai} + (1 - p_i) D_a$,
  where D_a = dummy impact for event *a*

1	2	3	4	d	
10	50	100	300	800	Impact
.1	.3	.25	.5	1	Raw success probability p_i
721	575	625	550	800	One-imperfect-witness impact d′

Methods we considered for solving impSP

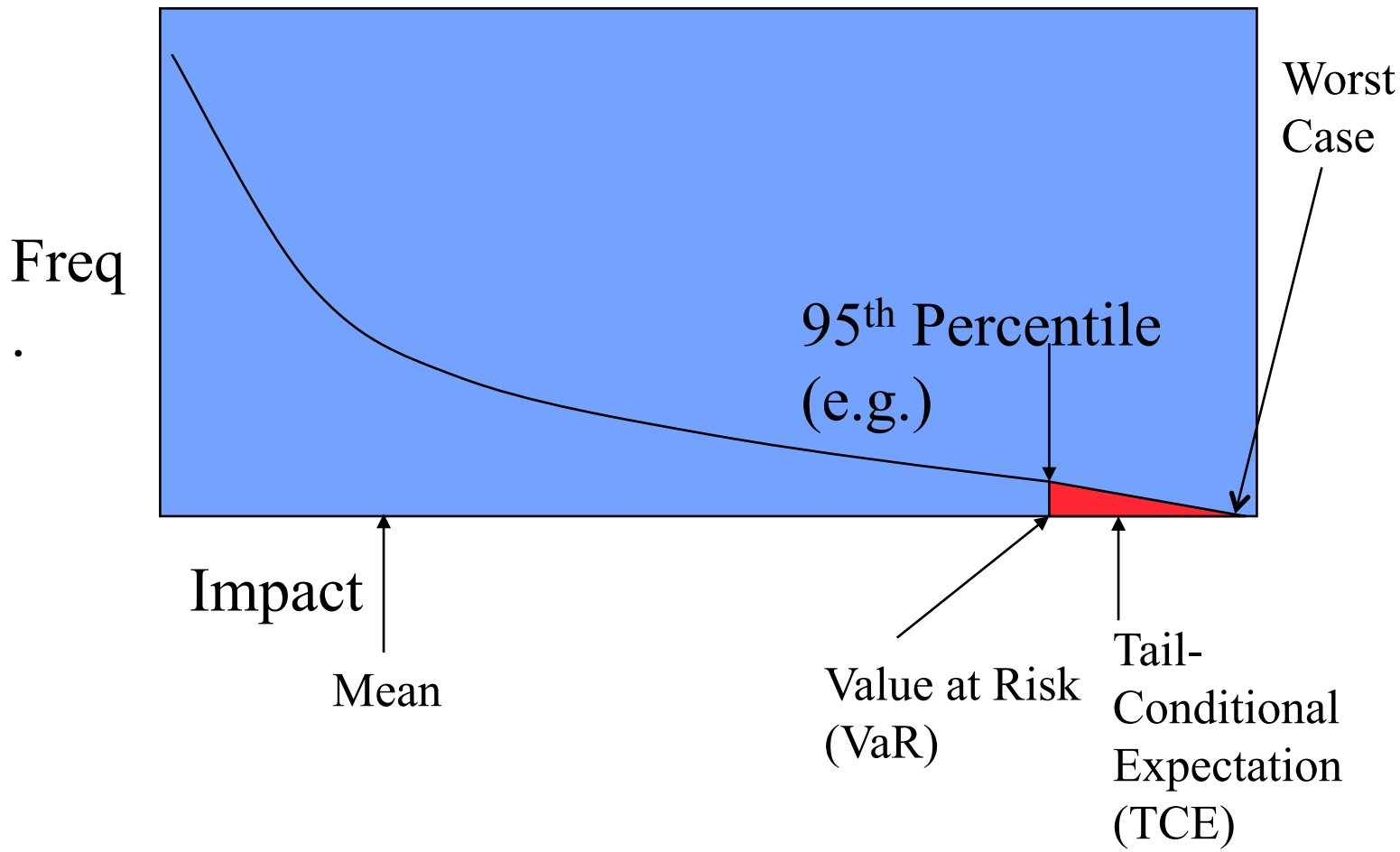
- Ignore imperfection
- Exact linear integer program based on zones
- Nonlinear solver (fractional)
- Local search with imperfect-sensor objective
- Random Sampling
- One-imperfect witness

11,575 nodes
9705 events
40 sensors



Sandia National Laboratories

Robust Scenario Coverage



- Robust (tail) measures typically harder than mean
- New method (for some cases) to find TCE using iterated mean

Multiple Objectives - Pareto Front

- Example: sensor network
  - # exposed/sickened/killed, mass released, pipe-feet contaminated, robust measures
- Represent each solution with a vector of objectives
- A solution dominates another if it's as least as good on all objectives:

$$(10,20) < (8,15)$$

- A solution is Pareto optimal if no other solution dominates it

$$(10,15) <> (8,20)$$

- Exploring Pareto front avoids value judgments
- Open research: Present decision maker with "small" set with
  - Objective diversity
  - Structural diversity

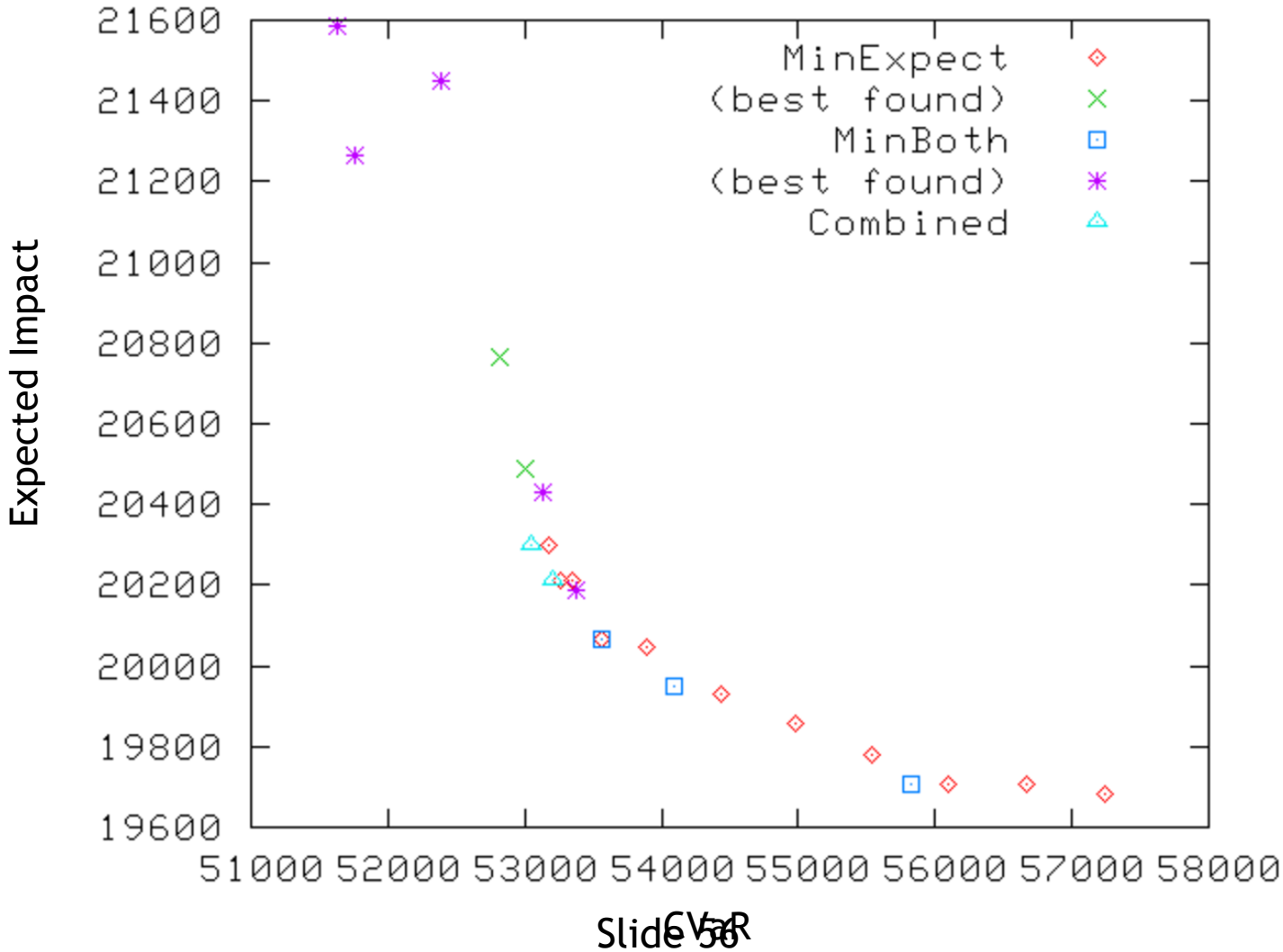
Multiple Objectives – Pareto Front

- Could consider using weights on multiple objectives:

  $\alpha w_1 + (1-\alpha) w_2$
- Can be difficult to solve, and doesn't always expose pareto-optimal solutions (convex hull)
- Goal constraints: bound one objective and optimize the other

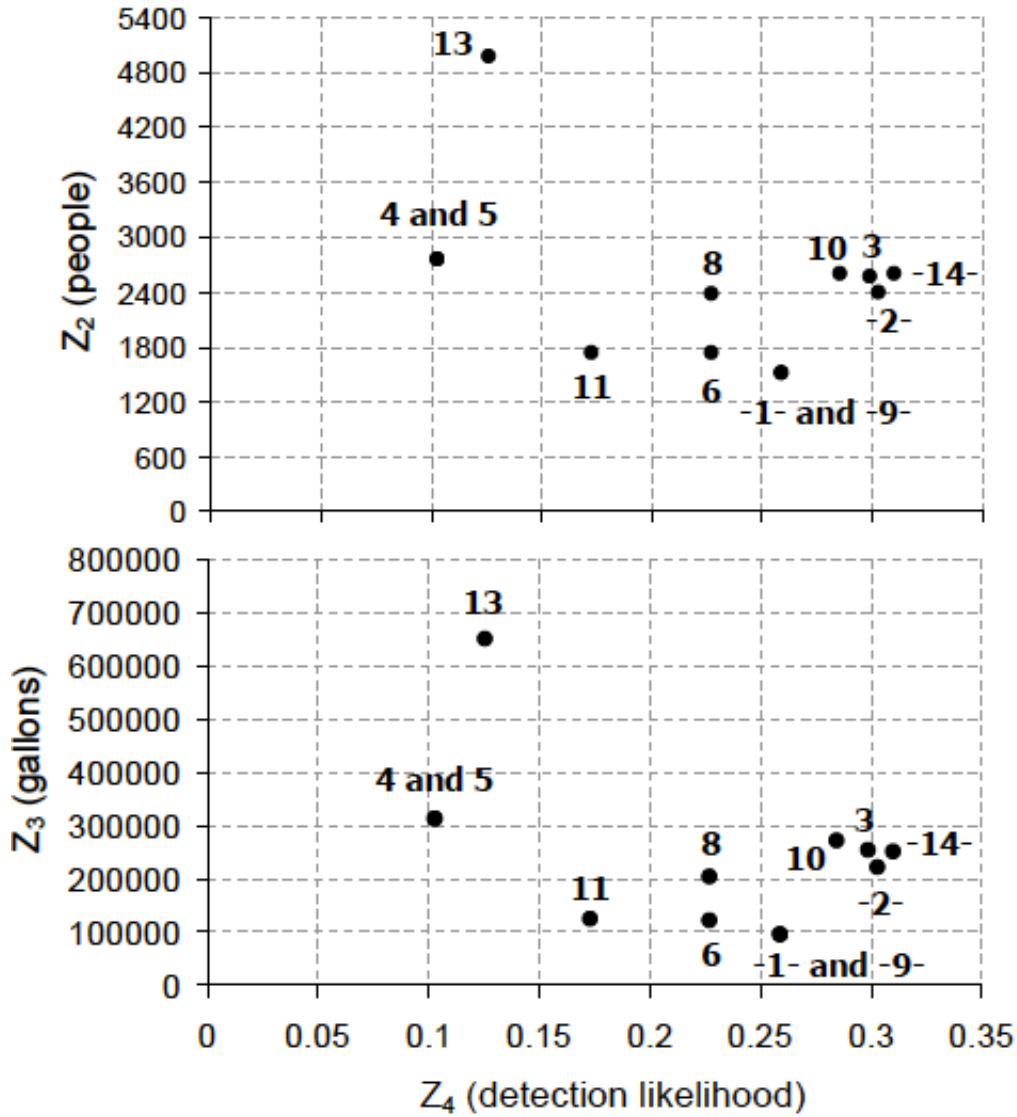
Multiobjective Example: Sensor Placement

Mean/CVaR (≈TCE) trade-off for Network B (3500 nodes)



Battle of the Water Sensor Networks

- Not a great example of experimentation but…



Sandia National Laboratories

Partnership: Optimizers and Domain Experts

- Optimizers
  - Model for performance
  - Have a “bag of tricks”
  - Generally know software availability or can roll own quickly
- Domain Experts
  - Model to solve a real problem
  - Want insight/understanding
  - Work with optimizers to ensure critical constraints are kept