Implicit Hitting Set Problems, Multi-Genome Alignment and Colorful Connected Subgraphs

Richard M. Karp

CPM

New York, June, 2010

Hitting Set Problem

- Input: a finite ground set U, a positive weight for each element, and a family Γ^* of subsets of U (circuits).
- Hitting set: a set H having a non-empty intersection with each circuit.
- Problem: find a hitting set of minimum weight.

Complexity of the Hitting Set Problem

- Equivalent to the weighted set cover problem
- NP-hard and hard to approximate within ratio o (log | Γ^*|).
- Greedy algorithm approximates within O(log | Γ^*|).
- In practice greedy algorithm gives good approximate solutions and optimal solutions can be computed fairly efficiently by a branch-and-cut algorithm.

Implicit Hitting Set Problem

- The collection of circuits has a compact implicit description
- A *separation oracle* is available which, given a subset H of the ground set, either determines that H is a hitting set or returns a circuit not hit by H.

Examples

- Feedback vertex set in a graph or digraph
- Feedback edge set in a digraph
- Max cut
- Intersection of k matroids
- Largest consistent subset of a set of linear inequalities
- *Multi-genome alignment*

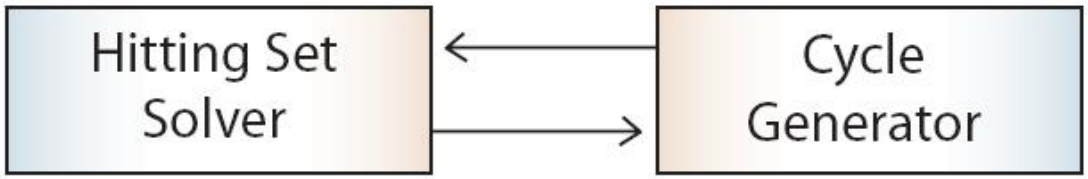
Separation Oracles

Efficient separation oracles exist for minimum feedback vertex set, minimum feedback arc set, max cut, k-matroid intersection.

Implicit Hitting Set Algorithm

- The implicit hitting set algorithm is a "dialogue" between a hitting set solver and a generator of circuits.
- Approach: build up a small set of critical circuits, solve the resulting explicit hitting set problem, and demonstrate that its optimal solution is feasible (and hence optimal) for the implicit hitting set problem.

Row Generation



A Simple but Inefficient Algorithm

$\Gamma \leftarrow$ empty set

Repeat:

$H \leftarrow$ optimal hitting set for Γ;

if H hits every circuit in Γ*

then return H and halt;

else add to Γ one or more circuits not hit by H

A Refinement

$\Gamma \leftarrow$ empty set

Repeat:

$H \leftarrow$ greedy hitting set for Γ;

if H hits every circuit in Γ *

then $H \leftarrow$ optimal hitting set for Γ;

if H hits every circuit in Γ*

then return H and halt;

Add to Γ one or more circuits not hit by H

Fractional Hitting Set Problem

- Min $c'x$ subject to $Ax <= 1$, $x >= 0$

  Rows of A are incidence vectors of sets in Γ^*

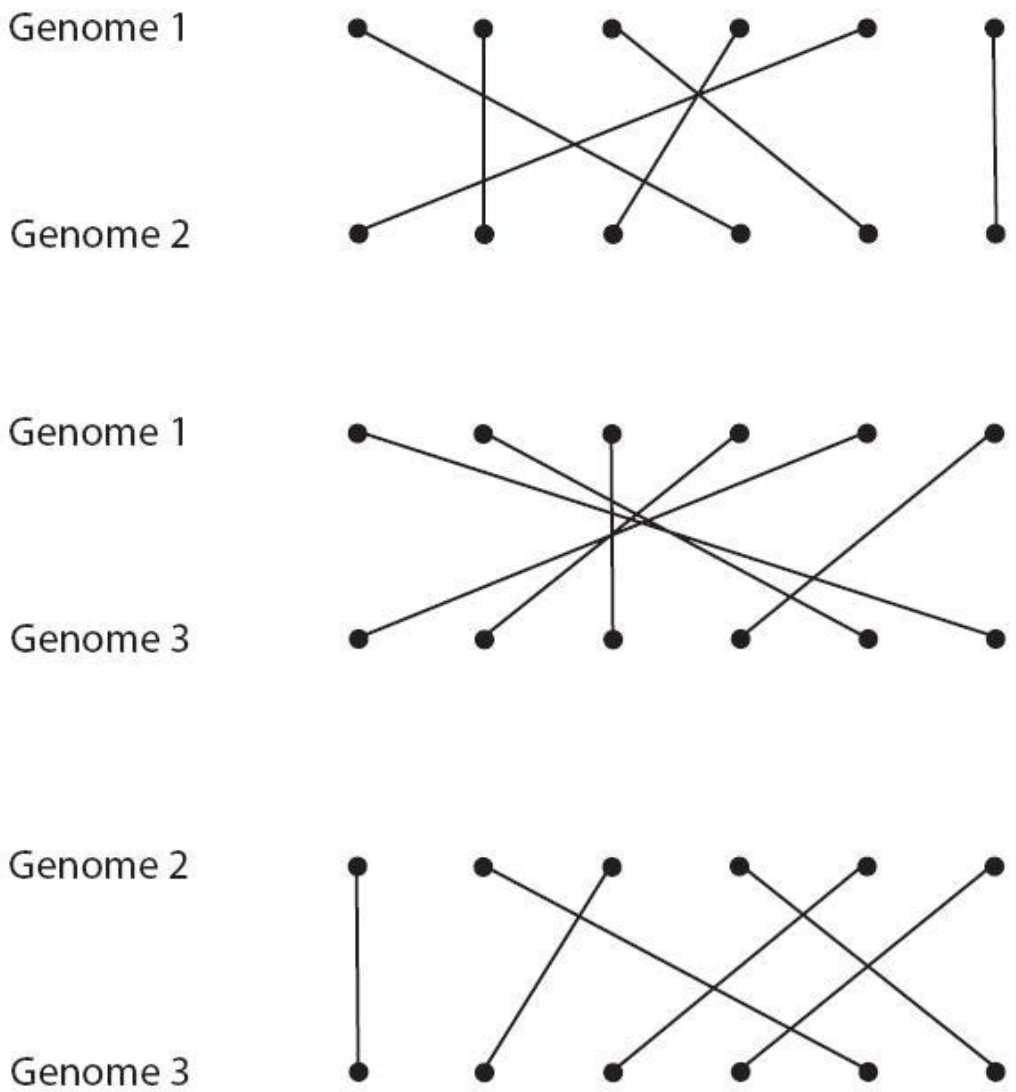
  x represents a fractional hitting set.

  Solvable efficiently by combination of linear programming with Lagrangian relaxation (Plotkin, Shmoys, Tardos)

Multi-Genome Alignment

- Input: the genomes of several closely related species.
- *Anchor*: a segment of one genome that matches a segment of another given genome.
- Each matching pair of anchors is assigned a weight.
- *Alignment*: assignment of a positive integer (column) to each anchor, such that, across each genome, the assignments strictly increase from left to right. Anchors are *aligned* if they are assigned the same integer.
- Problem: Find an alignment that maximizes the sum of the weights of the aligned anchor pairs.

3-Genome Alignment



Complexity Bounds

- The *2*-chain problem is equivalent to the maximum-weight increasing subsequence problem and is solvable in time $O(n \log n)$, where n is the cardinality of the ground set. The k-chain problem can be solved in time $O(n^k)$ by dynamic programming.

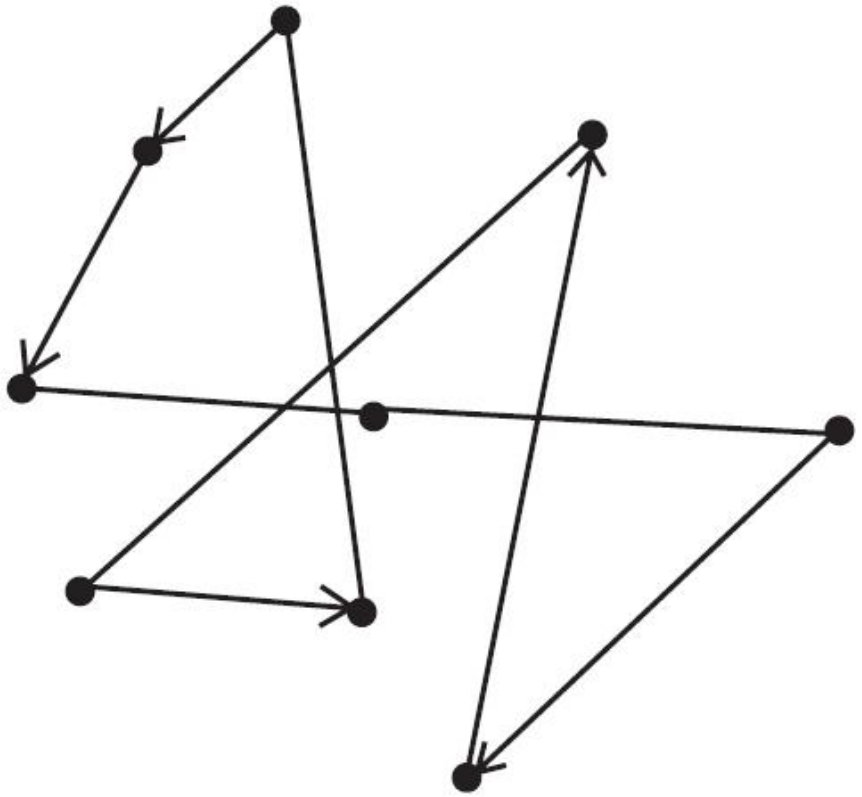
Graph-Theoretic Formulation

- A mixed graph with a vertex for each anchor.
- A directed edge from each anchor to the immediately following anchor in its genome.
- An undirected edge between two matching anchors, with a weight indicating the significance of the match.

Mixed Cycle

- A cycle is *mixed* if it contains at least one directed edge.
- A set of edges can be aligned if and only if it does not include all the undirected edges of any mixed cycle (Kececioglu). This leads to an integer programming formulation and a cutting-plane algorithm for the alignment problem (Kececioglu et al).

Mixed Cycle



Alignment as an Implicit Hitting Set Problem

- Ground set: set of undirected edges of the mixed graph (similar pairs of anchors).
- Hitting set: contains at least one undirected edge from each mixed cycle.
- The problem is implicit, since we don't wish to list all the mixed cycles.
- Generator of violated constraints: given a set *H*, we can either determine that it is a hitting set, or find a mixed cycle that *H* does not hit.

Solving the Alignment Problem

H : subset of ground set;

Γ : collection of circuits (undirected edge sets of mixed cycles)

$\Gamma \leftarrow$ empty set

Repeat:

$H \leftarrow$ greedy hitting set for Γ;

if H hits every circuit in Γ * *(every mixed cycle)*

then $H \leftarrow$ optimal hitting set for Γ;

if H hits every circuit in Γ* *(every mixed cycle)*

then return H and halt;

Add to Γ one or more mixed cycles not hit by H

Algorithmic Details

- Implementation of greedy algorithm
- The *CPLEX integer programming code* is used to find an optimal hitting set for Γ.
- Finding mixed cycles not hit by H:
- Delete edges in H.

  Method 1: depth-first search;

  Method 2: Attempt to construct an alignment column by column; whenever the construction is blocked, a mixed cycle is obtained.

Lower Bound

- For any set of circuits Γ, solve the LP relaxation of the explicit hitting set problem determined by Γ. The optimal value is a lower bound on the optimal value of the implicit hitting set problem.
- CPLEX yields stronger lower bounds

4096 Problems Derived from 5 Worm Genomes

Provably optimal solutions were obtained for 4055 of the 4096 problems. Summary:

(min time (sec.), max time, # probs., median #edges, Max # edges)

(0, 0.01, 1311, 52, 399)
(0.01, 0.1, 764, 203, 549)
(0, 1, 1086, 450, 1387)
(1, 10 6 32, 1104, 4645)
(10, 60, 151, 1351, 12313)
(60, 600, 75, 1136, 14690)
(600, 3600 , 36, 1236, 13916)

Tuning the Algorithm

- In the multi-genome alignment problem there are many choices of methods for choosing a near-optimal or optimal hitting set H given Γ, and for generating a set of mixed cycles not hit by H. A systematic hillclimbing search through the "space" of these choices was required to obtain good performance.

Methodology for Heuristic Algorithm Design

- Components:
  (1) Algorithmic strategy
  (2)Training set of instances
  (3)Tuning algorithm for selection of optimal variant of the algorithm
  (4) Validation process

Example: Colorful Subgraph Problem

- Input: a graph, and an assignment of a color to each vertex
- Problem: Find (if possible) a connected subgraph containing exactly one vertex of each color.
- Has been attacked by dynamic programming and branch-and-cut integer programming.

Heuristic Algorithmic Strategy

- Repeat:

  (1) Delete vertices with frequent colors;

  (2) In the remaining graph, select a minimal set of connected components covering all infrequent colors;

  (3) Insert minimal set of vertices with frequent colors to restore connectedness and cover all colors;

  (4) Delete redundant degree-1 vertices.

Example on Grid

W E L C O M E T
O T H E W E B S
I T E O F T H E
A N N U A L S Y
M P O S I U M O
N C O M B I N A
T O R I A L P A
T T E R N M A T

Delete Frequent Letters (E,T,A, O,M)

W L C

H W B S

I F H

N N U L S Y

P S I U

N C B I N

R I L P

R N

Delete Redundant Components

W B S
F H
N N U L S Y
P S I U
N C
R I
R N

Reconnect and Cover Deleted Letters

W E B S

F T H

N N U A L S Y

P S I U

- N C O M
- R I
- R N

Delete Redundant Degree-1 Nodes

W E B

F T H

U A L S Y

P S I U

C O M

R I

R N

Delete Frequent Letters (U,S,I,R)

W E B

F T H

A L Y

P

C O M

N

Reconnect and Cover Frequent Letters

- W E B
- F T H
- A L S Y
- P S I
- COM B
- A
- R N

Delete Frequent Letters (B,A,S)

W E

F T H

L Y

P I U

C OM

R N

Reconnect and Cover Deleted Letters

- W E
- F T H
- L S Y
- P I U
- C O M B
- A
- R N

Implicit Optimization Problems

- Set of constraints defined implicitly by a generation algorithm rather than by an explicit list.

  -- Linear and convex programming: equivalence of separation and optimization

  -- Integer programming: cutting-plane methods

  -- Linear programming: column generation

Equivalence of Separation and Optimization (GLS)

Minimize $c'x$

subject to x in convex set K

- K described by a separation oracle:

  given point x, oracle either asserts that x in K or returns hyperplane separating x from K. This suffices for polynomial-time algorithm.

Cutting Plane Methods for Integer Programming

- Minimize $c'x$ subject to $Ax <= b$,

  x integer
- Repeat:

  Solve LP relaxation. If optimal point x is fractional, add constraints satisfied by all integer solutions but violated by x.

Column Generation in LP

- Variables (columns of LP) not explicitly given; number of variables may be huge.
- Auxiliary algorithm generates new column to enter basis.
- Example: cutting-stock problem. Auxiliary algorithm solves a knapsack problem.
- Row generation: auxiliary algorithm generates rows (constraints) on demand.