CSE 421: Algorithms

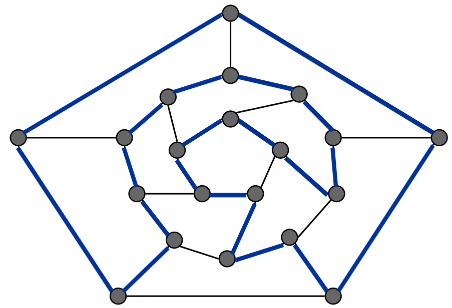
Winter 2014

Lecture 24-25: Poly-time reductions

Reading:
Sections 8.4-8.8

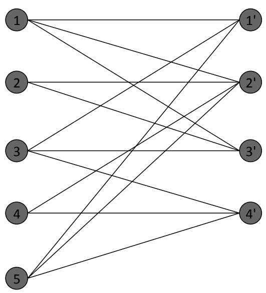
hamiltonian cycle

- **HAM-CYCLE:** given an undirected graph G = (V, E), does there exist a simple cycle Γ that contains every node in V.



hamiltonian cycle

- **HAM-CYCLE:** given an undirected graph G = (V, E), does there exist a simple cycle Γ that contains every node in V.

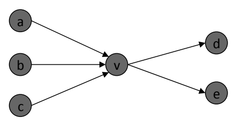


NO: bipartite graph with odd number of nodes.

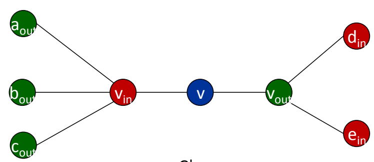
directed hamiltonian cycle

- DIR-HAM-CYCLE: given a digraph G = (V, E), does there exists a simple directed cycle Γ that contains every node in V?

- Claim. DIR-HAM-CYCLE $\leq_P$ HAM-CYCLE.
- Pf. Given a directed graph G = (V, E), construct an undirected graph G' with 3n nodes.



G



G'

directed hamiltonian cycle

- Claim: G has a Hamiltonian cycle iff G' does.

- Pf. $\Rightarrow$
  - Suppose G has a directed Hamiltonian cycle Γ.
  - Then G' has an undirected Hamiltonian cycle (same order).

- Pf. $\Leftarrow$
  - Suppose G' has an undirected Hamiltonian cycle Γ'.
  - Γ' must visit nodes in G' using one of following two orders:

    ..., B, G, R, B, G, R, B, G, R, B, ...

    ..., B, R, G, B, R, G, B, R, G, B, ...
  - Blue nodes in Γ' make up directed Hamiltonian cycle Γ in G, or reverse of one. ▪

3-SAT $\leq_P$ DIR-HAM-CYCLE

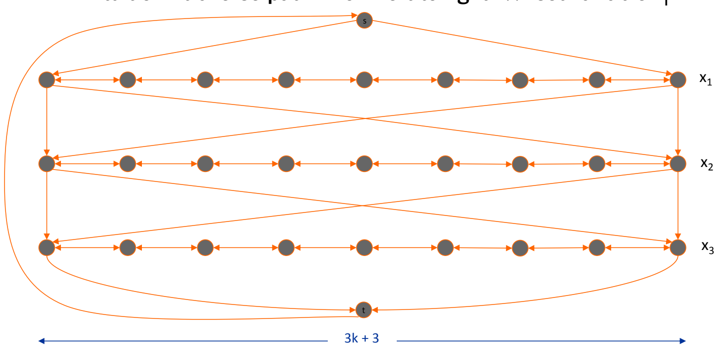
- Claim: 3-SAT $\leq_P$ DIR-HAM-CYCLE.

- Pf. Given an instance Φ of 3-SAT, we construct an instance of DIR-HAM-CYCLE that has a Hamiltonian cycle iff Φ is satisfiable.

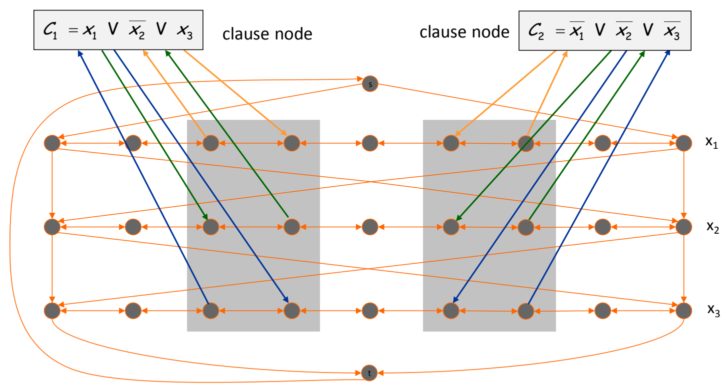
- Construction. First, create graph that has 2^n Hamiltonian cycles which correspond in a natural way to 2^n possible truth assignments.

3-SAT $\leq_P$ DIR-HAM-CYCLE

- Construction. Given 3-SAT instance Φ with n variables x_i and k clauses.
  - Construct G to have 2^n Hamiltonian cycles.
  - Intuition: traverse path i from left to right $\Leftrightarrow$ set variable $x_i = 1$.



3-SAT $\leq_P$ DIR-HAM-CYCLE



3-SAT $\leq_P$ DIR-HAM-CYCLE

- **Claim: Φ is satisfiable iff G has a Hamiltonian cycle.**

- **Pf. $\Rightarrow$**
  - **Suppose 3-SAT instance has satisfying assignment x*.**
  - **Then, define Hamiltonian cycle in G as follows:**
    - if $x^*_i = 1$, traverse row i from left to right
    - if $x^*_i = 0$, traverse row i from right to left
    - for each clause C_j, there will be at least one row i in which we are going in "correct" direction to splice node C_j into tour

3-SAT $\leq_P$ DIR-HAM-CYCLE

- **Pf. $\Leftarrow$**
  - **Suppose G has a Hamiltonian cycle Γ.**
  - **If Γ enters clause node C_j, it must depart on mate edge.**
    - thus, nodes immediately before and after C_j are connected by an edge e in G
    - removing C_j from cycle, and replacing it with edge e yields Hamiltonian cycle on G - { C_j }
  - **Continuing in this way, we are left with Hamiltonian cycle Γ' in G - { C_1, C_2, . . . , C_k }.**
  - **Set $x^*_i = 1$ iff Γ' traverses row i left to right.**
  - **Since Γ visits each clause node C_j, at least one of the paths is traversed in "correct" direction, and each clause is satisfied.** ▪

longest path

- **SHORTEST-PATH. Given a digraph G = (V, E), does there exists a simple path of length at most k edges?**

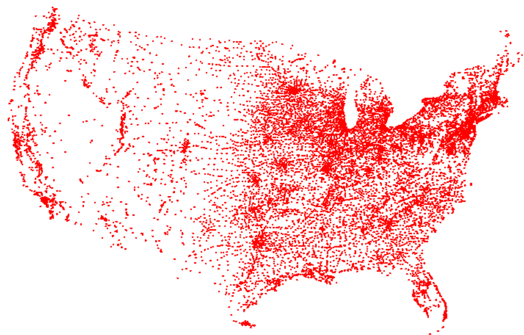
- **LONGEST-PATH. Given a digraph G = (V, E), does there exists a simple path of length at least k edges?**

- **Claim. 3-SAT $\leq_P$ LONGEST-PATH.**

- **Pf 1. Redo proof for DIR-HAM-CYCLE, ignoring back-edge from t to s.**
- **Pf 2. Show HAM-CYCLE $\leq_P$ LONGEST-PATH.**

traveling salesperson problem

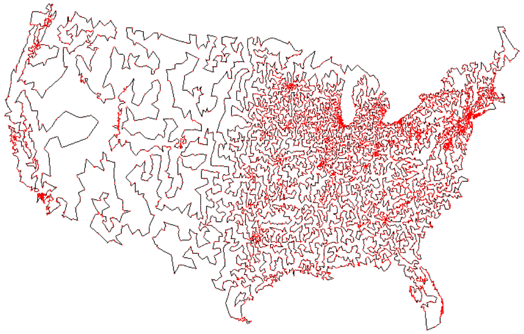
- **TSP. Given a set of n cities and a pairwise distance function d(u, v), is there a tour of length $\leq$ D?**



All 13,509 cities in US with a population of at least 500
Reference: http://www.tsp.gatech.edu

traveling salesperson problem

- **TSP.** Given a set of n cities and a pairwise distance function d(u, v), is there a tour of length ≤ D?



Optimal TSP tour
Reference: http://www.tsp.gatech.edu

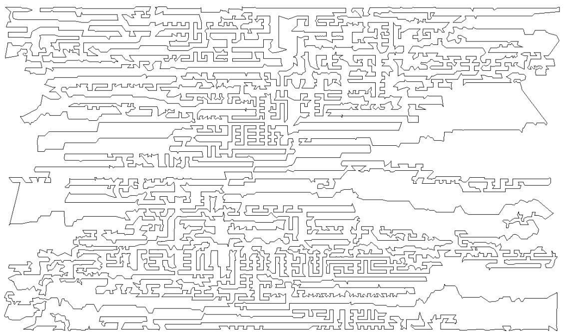
traveling salesperson problem

- **TSP.** Given a set of n cities and a pairwise distance function d(u, v), is there a tour of length ≤ D?

11,849 holes to drill in a programmed logic array
Reference: http://www.tsp.gatech.edu

traveling salesperson problem

- **TSP.** Given a set of n cities and a pairwise distance function d(u, v), is there a tour of length ≤ D?



Optimal TSP tour
Reference: http://www.tsp.gatech.edu

3-dimensional matching

- **3D-MATCHING.** Given n instructors, n courses, and n times, and a list of the possible courses and times each instructor is willing to teach, is it possible to make an assignment so that all courses are taught at different times?

Instructor	Course	Time
Wayne	COS 423	MW 11-12:20
Wayne	COS 423	TTh 11-12:20
Wayne	COS 226	TTh 11-12:20
Wayne	COS 126	TTh 11-12:20
Tardos	COS 523	TTh 3-4:20
Tardos	COS 423	TTh 11-12:20
Tardos	COS 423	TTh 3-4:20
Kleinberg	COS 226	TTh 3-4:20
Kleinberg	COS 226	MW 11-12:20
Kleinberg	COS 423	MW 11-12:20

3-dimensional matching

- **3D-MATCHING.** Given disjoint sets X, Y, and Z, each of size n and a set $T \subseteq X \times Y \times Z$ of triples, does there exist a set of n triples in T such that each element of $X \cup Y \cup Z$ is in exactly one of these triples?

- **Claim. 3-SAT $\leq_P$ 3D-Matching.**
- Pf. Given an instance Φ of 3-SAT, we construct an instance of 3D-matching that has a perfect matching iff Φ is satisfiable.

3-dimensional matching

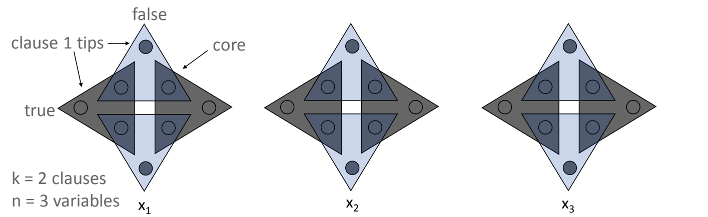
Construction. (part 1)

– Create gadget for each variable x_i with 2k core and tip elements.

number of clauses

– No other triples will use core elements.

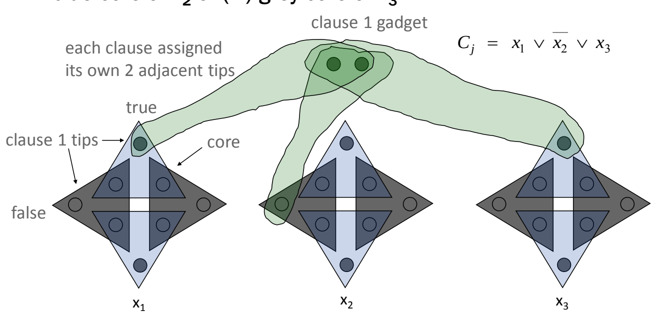
– In gadget i, 3D-matching must use either both **grey** triples or both **blue** ones.



3-dimensional matching

Construction. (part 2)

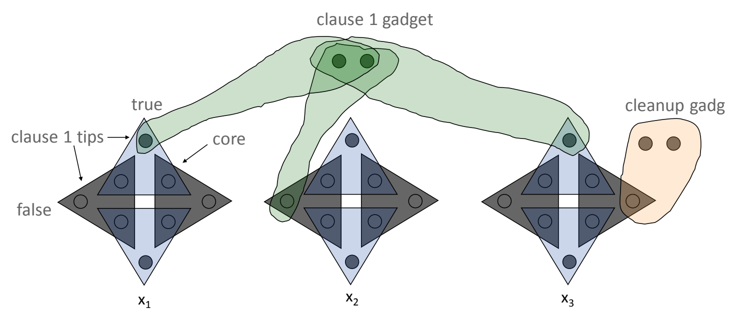
- For each clause C_j create two elements and three triples.
- Exactly one of these triples will be used in any 3D-matching.
- Ensures any 3D-matching uses either (i) grey core of x_1 or (ii) blue core of x_2 or (iii) grey core of x_3.



3-dimensional matching

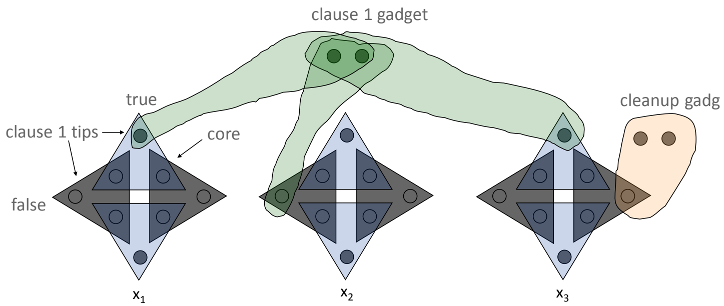
Construction. (part 3)

For each tip, add a cleanup gadget.



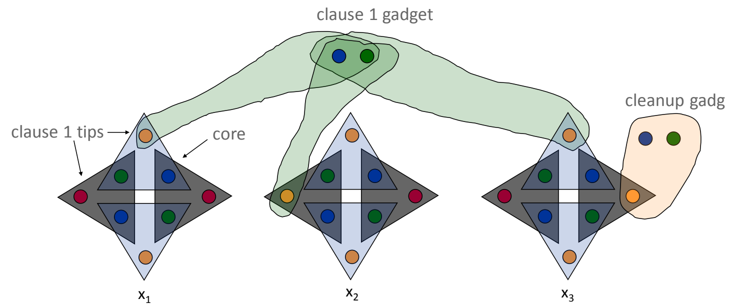
3-Dimensional Matching

- **Claim.** Instance has a 3D-matching iff Φ is satisfiable.
- **Detail.** What are X, Y, and Z? Does each triple contain one element from each of X, Y, Z?



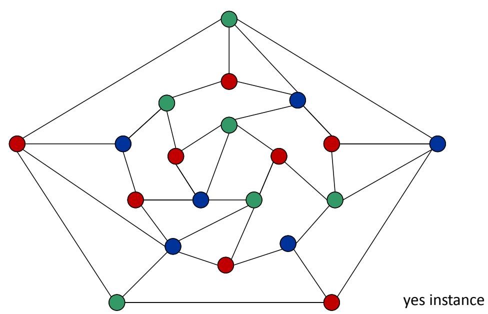
3-Dimensional Matching

- **Claim.** Instance has a 3D-matching iff Φ is satisfiable.
- **Detail.** What are X, Y, and Z? Does each triple contain one element from each of X, Y, Z?



3-colorability

- **3-COLOR:** Given an undirected graph G does there exists a way to color the nodes red, green, and blue so that no adjacent nodes have the same color?



register allocation

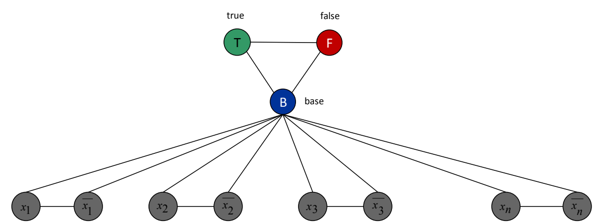
- **Register allocation.** Assign program variables to machine register so that no more than k registers are used and no two program variables that are needed at the same time are assigned to the same register.
- **Interference graph.** Nodes are program variables names, edge between u and v if there exists an operation where both u and v are "live" at the same time.
- Observation. [Chaitin 1982] Can solve register allocation problem iff interference graph is k-colorable.
- 3-COLOR $\le_P$ k-REGISTER-ALLOCATION for any constant $k \ge 3$.

3-colorability

- Claim. 3-SAT $\leq_P$ 3-COLOR.
- Pf. Given 3-SAT instance Φ, we construct an instance of 3-COLOR that is 3-colorable iff Φ is satisfiable.
- Construction.
  i. For each literal, create a node.
  ii. Create 3 new nodes T, F, B; connect them in a triangle, and connect each literal to B.
  iii. Connect each literal to its negation.
  iv. For each clause, add gadget of 6 nodes and 13 edges.

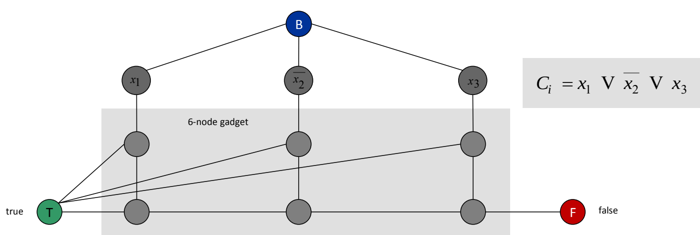
3-colorability

- Claim. Graph is 3-colorable iff Φ is satisfiable.
- Pf. $\Rightarrow$ Suppose graph is 3-colorable.
  - Consider assignment that sets all T literals to true.
  - (ii) ensures each literal is T or F.
  - (iii) ensures a literal and its negation are opposites.



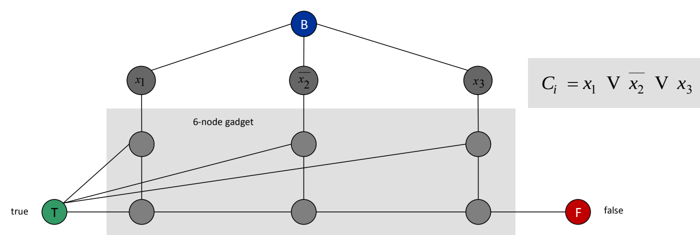
3-colorability

- Claim. Graph is 3-colorable iff Φ is satisfiable.
- Pf. $\Rightarrow$ Suppose graph is 3-colorable.
  - Consider assignment that sets all T literals to true.
  - (ii) ensures each literal is T or F.
  - (iii) ensures a literal and its negation are opposites.
  - (iv) ensures at least one literal in each clause is T.



3-colorability

- Claim. Graph is 3-colorable iff Φ is satisfiable.
- Pf. $\Rightarrow$ Suppose graph is 3-colorable.
  - Consider assignment that sets all T literals to true.
  - (ii) ensures each literal is T or F.
  - (iii) ensures a literal and its negation are opposites.
  - (iv) ensures at least one literal in each clause is T.



3-colorability

- Claim. Graph is 3-colorable iff Φ is satisfiable.
- Pf. $\Leftarrow$ Suppose 3-SAT formula Φ is satisfiable.
  - Color all true literals T.
  - Color node below green node F, and node below that B.
  - Color remaining middle row nodes B.
  - Color remaining bottom nodes T or F as forced. ▪