Color Coding

Speeding up Network Searches

858L

Efficient Algorithms for Detecting Signaling Pathways in Protein Interaction Networks

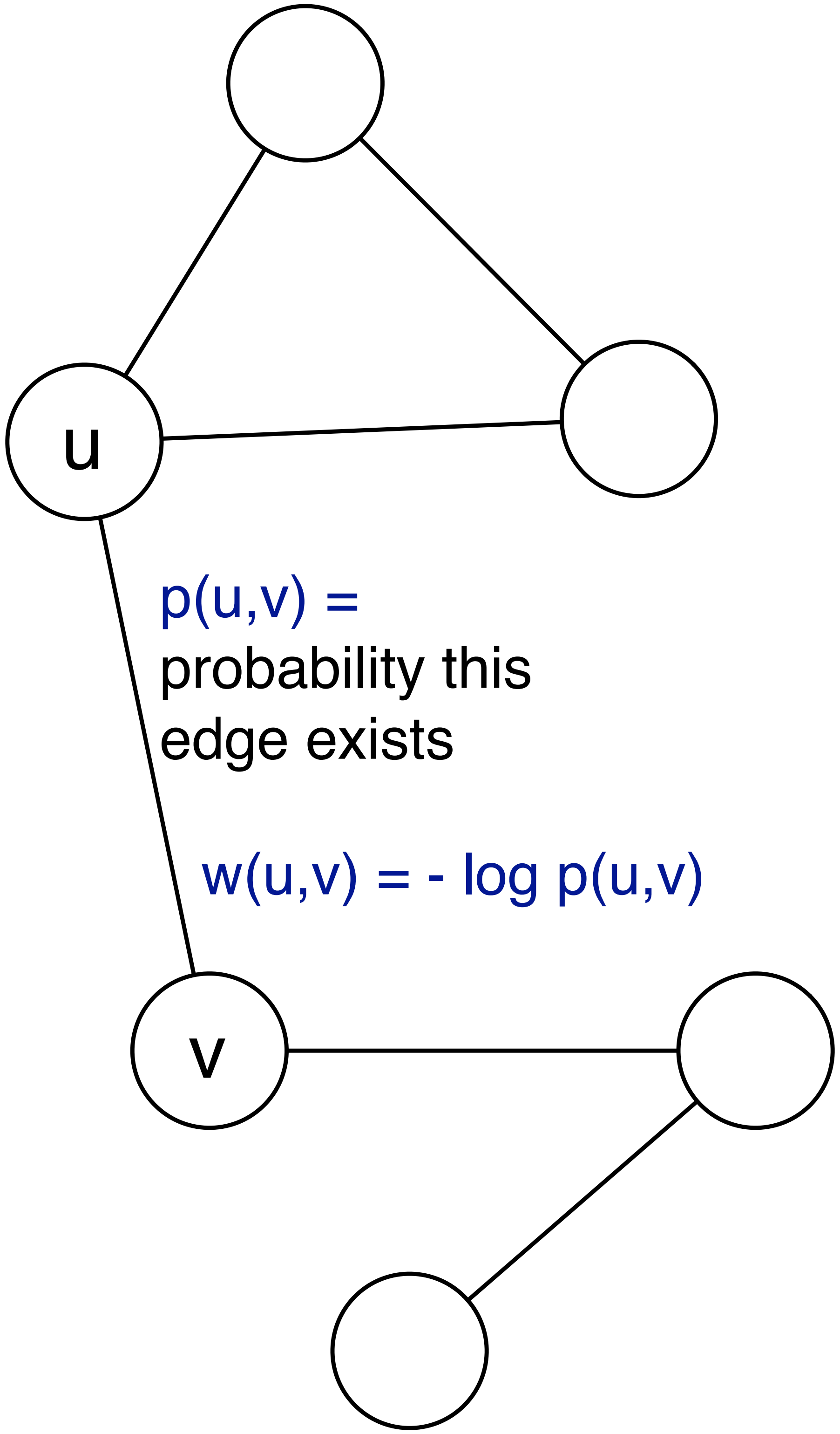
Scott, Ideker, Karp, Sharan

RECOMB 2005

- Color Coding: Alon et al, 1995.

Searching for High Scoring Paths

Weighted network G:



G might be an alignment graph, a PPI network, metabolic network, etc...

P = simple path

Weight(P) = sum of *w(u,v)* values along its edges

Length(P) = number of nodes in P

Goal: Low-weight, simple, length-k paths

Given: Graph G, a subset of nodes I, and a node v.

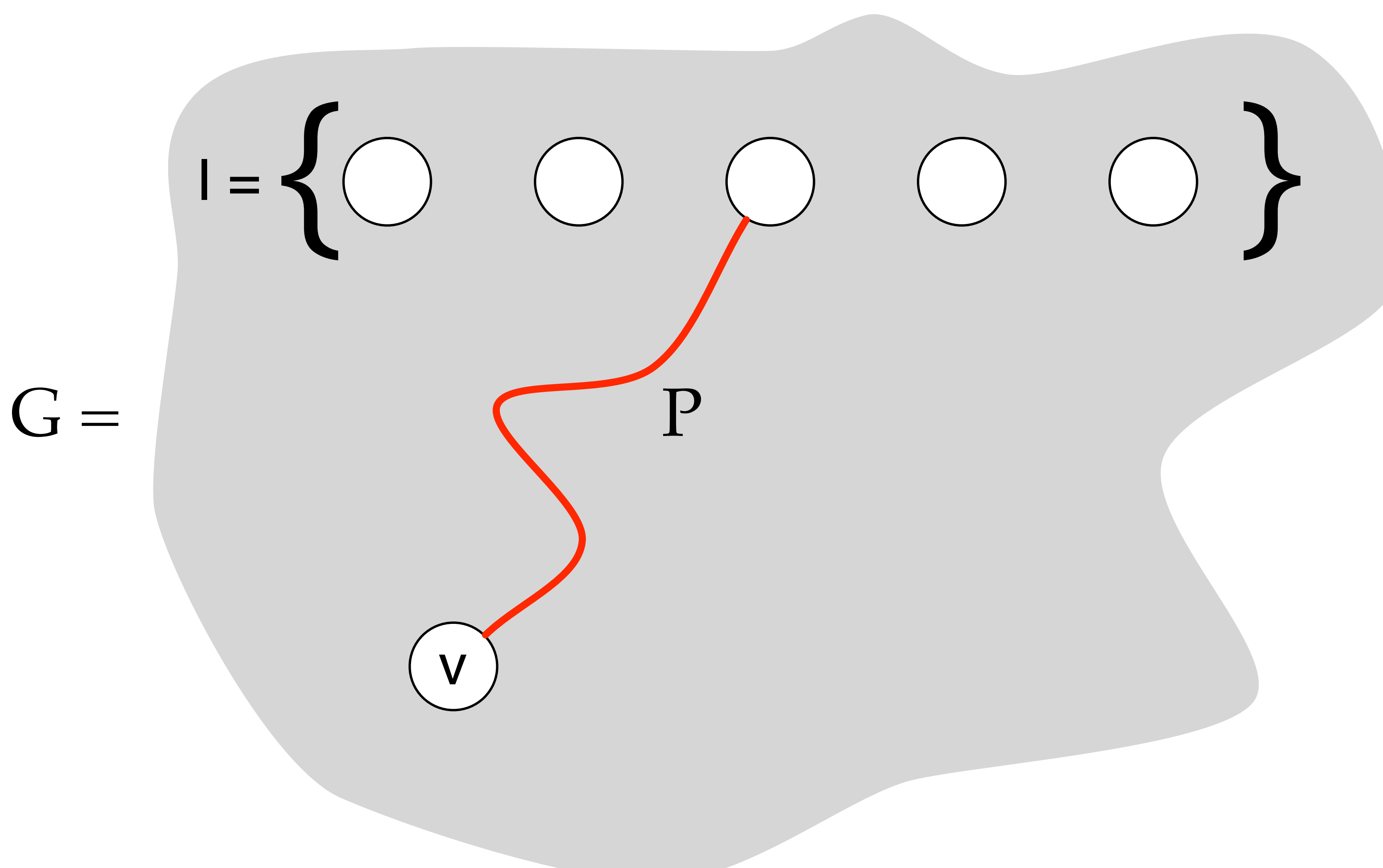
Find: The lowest-weight path P that:

(1) starts at some vertex in I

(2) ends at v

(3) is of length k and is simple (doesn't use any vertex twice)

Set I let's us specify, e.g., that the path should start at a surface receptor protein.



Is this Problem Hard?

Given: Graph G, a subset of nodes I, and a node v.

Find: The lowest-weight, simple, length-k path between I and v.

Is this Problem Hard?

Given: Graph G, a subset of nodes I, and a node *v*.
Find: The lowest-weight, simple, length-*k* path between I and v.

Yes. It's NP-hard. Why?

Is this Problem Hard?

Given: Graph G, a subset of nodes I, and a node v.
Find: The lowest-weight, simple, length-k path between I and v.

Yes. It's NP-hard. Why?

Reduce Hamiltonian Cycle (HC) to it: To solve an HC instance <G_H>, let $G = G_H$, $I = \{v\}$, and $k = n$.

Is this Problem Hard?

Given: Graph G, a subset of nodes I, and a node v.
Find: The lowest-weight, simple, length-k path between I and v.

Yes. It's NP-hard. Why?

Reduce Hamiltonian Cycle (HC) to it: To solve an HC instance <G_H>, let $G = G_H$, $I = \{v\}$, and $k = n$.

Without the simple condition or length-k condition, the problem is easy.

Dynamic Programming Algorithm

v ∈ S   Set of ≤ k vertices

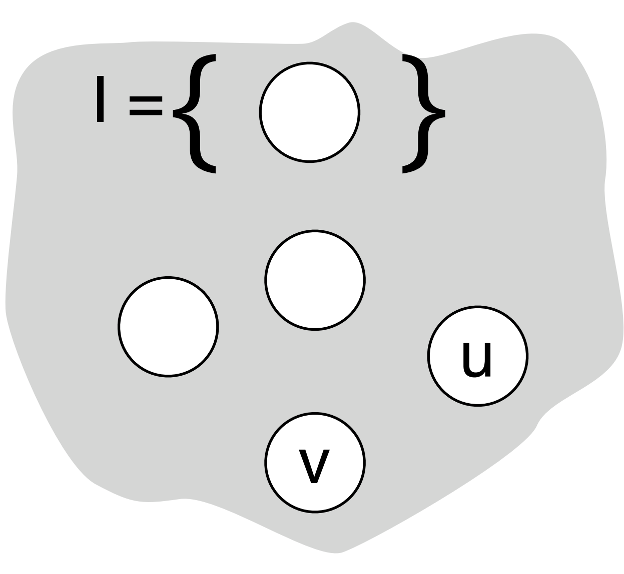
W(v, S) := minimum weight of a simple path that starts at I, visits each vertex in S, and ends at v, and is of length |S|.

W(v, S) := ∞ if no such path exists.

$$W(v, \{v\}) = \begin{cases} 0 & \text{if } v \in I \\ \infty & \text{if } v \notin I \end{cases}$$

$$W(v, S) = \min_{u \in S - \{v\}} W(u, S - \{v\}) + w(u, v)$$

Smaller size "S" set, so we can compute W(•, •) in order of increasing size of S.



Ok, So:

$$\mathrm{OPT}(I, v) = \min_{S:|S|=k} W(v, S)$$

Note how “simple” this algorithm is: try all possible sets of *k* nodes, compute their optimal order, and return the best set.

What’s the running time?

Ok, So:

$$\mathrm{OPT}(I, v) = \min_{S:|S|=k} W(v, S)$$

Note how "simple" this algorithm is: try all possible sets of *k* nodes, compute their optimal order, and return the best set.

What's the running time?

Number of sets we will consider = all possible subsets of nodes of size ≤ k =

$$\sum_{i=0}^{k} \binom{n}{i} = n^k$$

For each set, computing the min takes at most O(k) steps.

Therefore: Running time = $O(kn^k)$.

Color Coding

- $O(kn^k)$ is too slow for any interesting k.

- Can we do better?

- Idea: rather than keeping track of all of S, we'll keep track of less information about which nodes we've already visited.

- This will introduce a problem: we may miss the optimum path...

Color Coding

Main Step: Randomly color each node with a color from $\{1,2,\ldots,k\}$. Let $c(u)$ be the color of node u.

Define: a path is "colorful" if it contains exactly 1 vertex of each color.

Note: any colorful path is simple.

So, we consider this modified problem:

Given: Graph G, a subset of nodes I, and a node v.
Find: The lowest-weight, colorful, length-k path between I and v.

Color Coding DP Algorithm

$c(v) \in C$ Set of $\leq k$ colors

$\overline{W}(v, C)$:= minimum weight of a path that starts at I, visits a vertex of each color in C, ends at v, and is of length $|C|$.

$\overline{W}(v, C) := \infty$ if no such path exists.

$$\bar{W}(v, C) = \min_{u:c(u) \in C - \{c(v)\}} \bar{W}(u, C - \{c(v)\}) + w(u, v)$$

"C" keeps track of the remaining allowed colors.

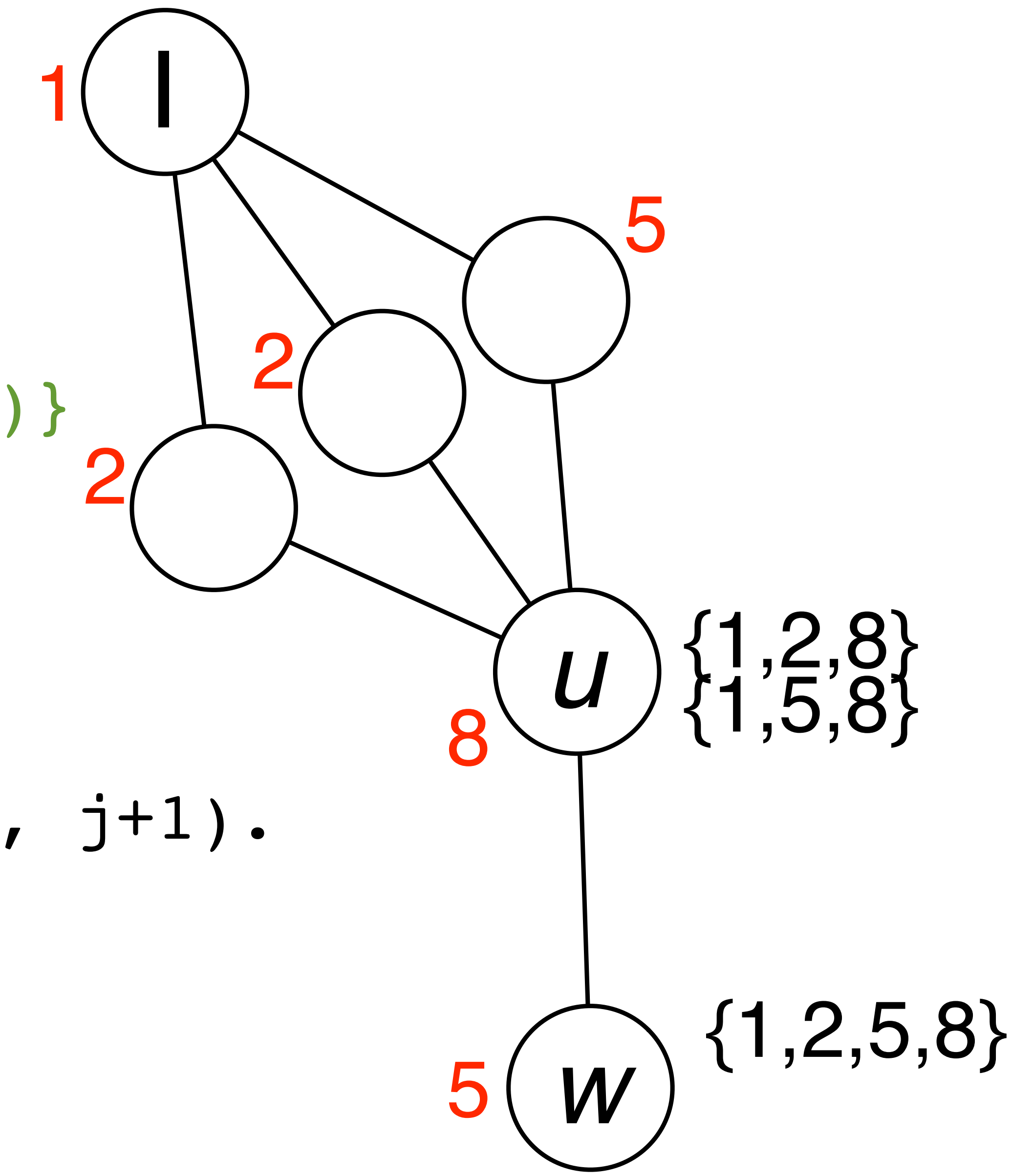
Intuition for faster run time: we must consider only 2^k possible sets "C" instead of $O(n^k)$

$$\sum_{i=0}^{k} \binom{k}{i} = 2^k$$

Alternative View of Color Coding Algorithm

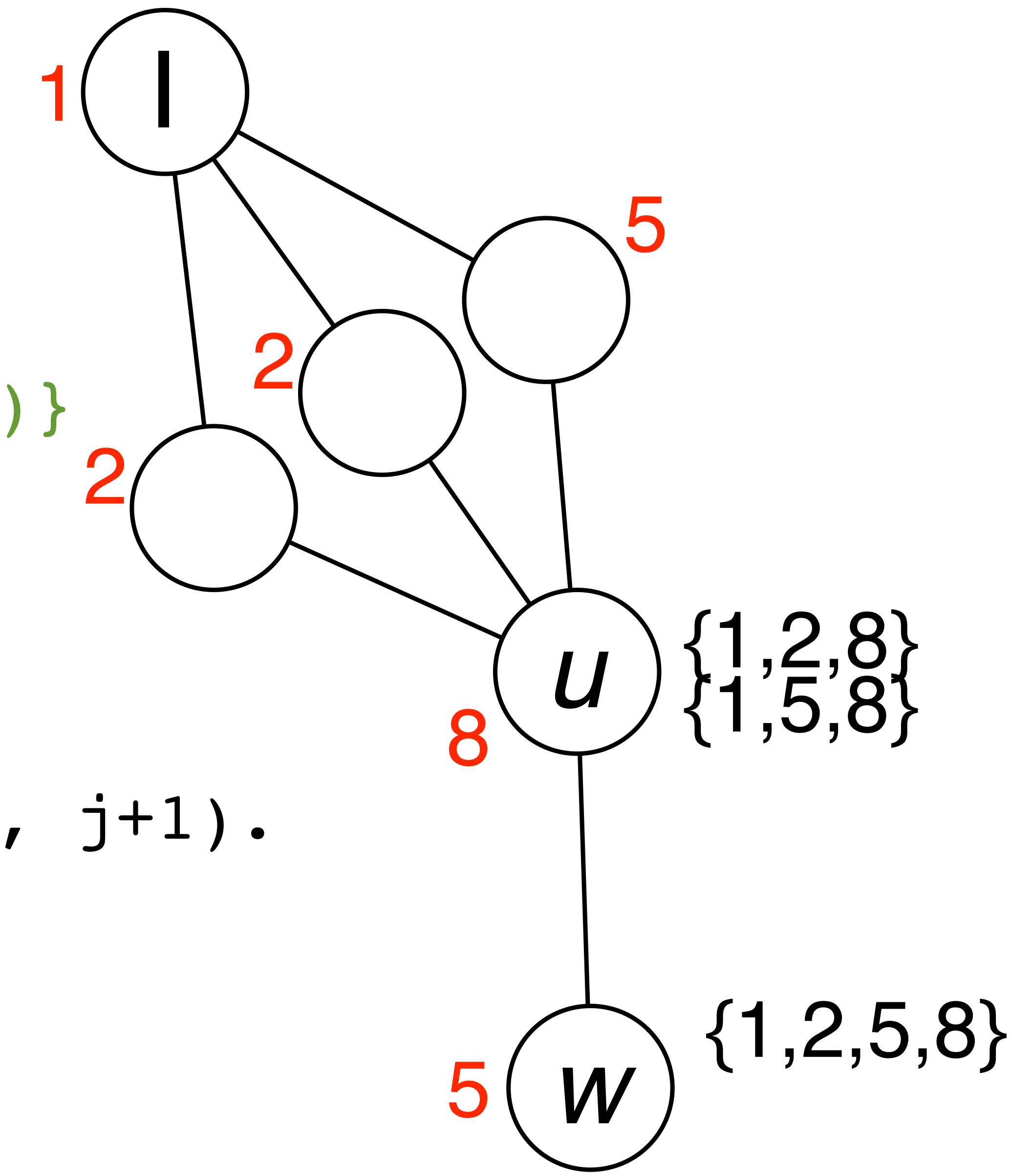
Let *I* be the given *starting* node set
Let colorings(u, j) be the set of valid path colorings for a path of length j-1 from *I* to u

```
For all u in I: colorings(u,1) = {c(u)}
For j = 1, ..., k:
  For every edge (u, w):
    For every C in colorings(u, j):
      If c(w) not in C:
        Add C U {c(w)} to colorings(w, j+1).
```



Alternative View of Color Coding Algorithm

Let I be the given *starting* node set
Let colorings(u, j) be the set of valid path colorings for a path of length j-1 from I to u



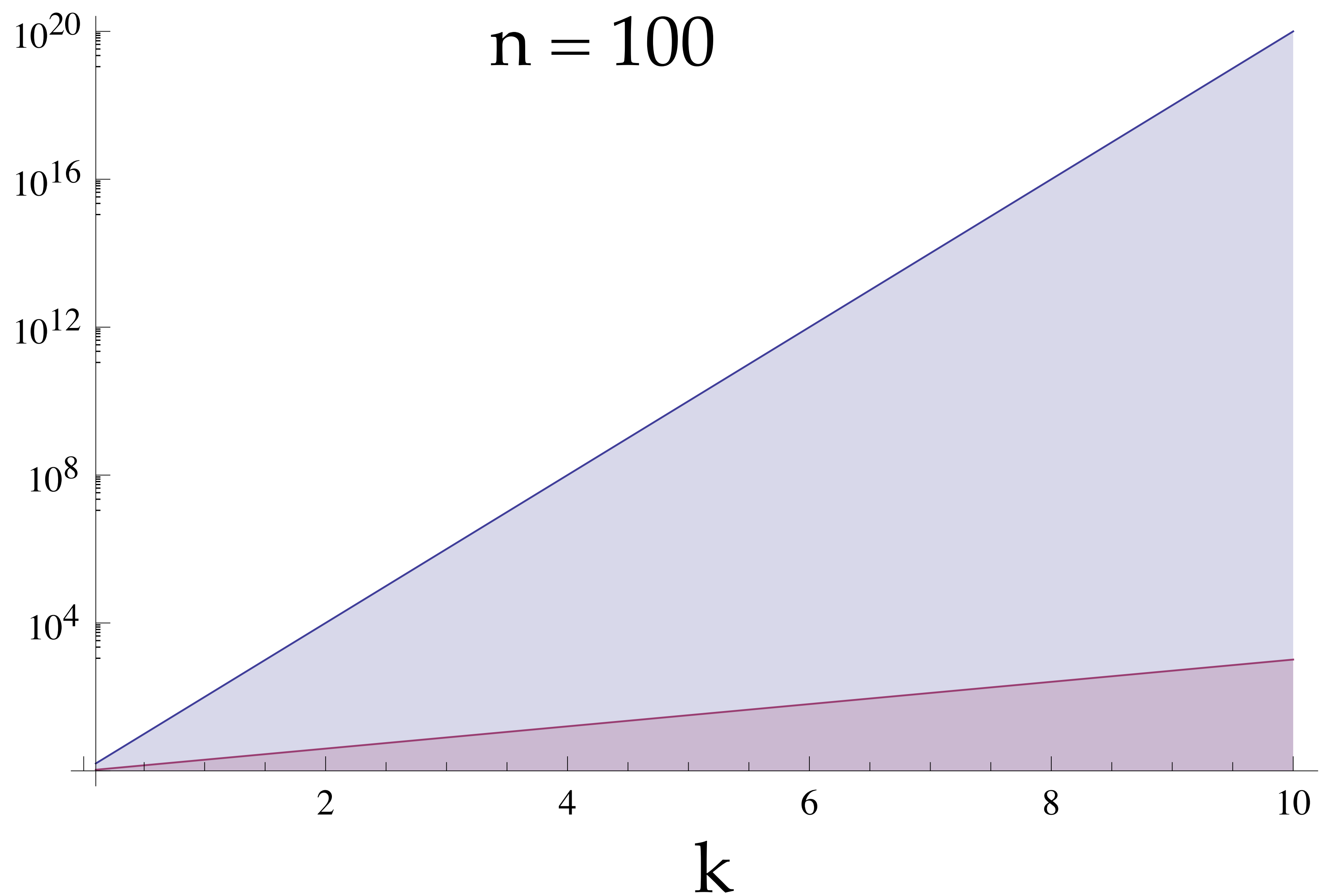

```
For all u in I: colorings(u,1) = {c(u)}
For j = 1, ..., k:
  For every edge (u, w):
    For every C in colorings(u, j):
      If c(w) not in C:
        Add C U {c(w)} to colorings(w, j+1).
```

Running time:

$$\sum_{j=0}^{k}\left[|E|\binom{k}{j}j\right] = O(2^k k|E|)$$

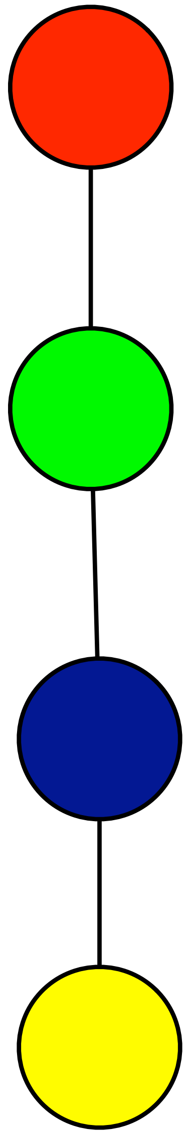
So:

We had an algorithm that was ≈ $O(n^k)$
We converted it into an ≈ $O(2^k)$ algorithm,
but with an ε probability we'll miss the optimal answer.



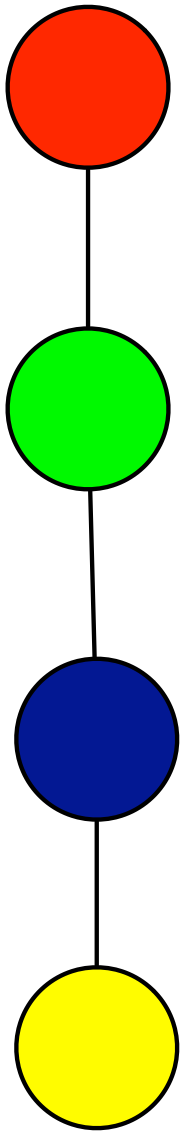
What if the optimal path is not colorful?

Have to repeat this procedure enough times so that the probability that that happens is low.



What if the optimal path is not colorful?

Have to repeat this procedure enough times so that the probability that that happens is low.



k! ways to make a path colorful.

k^k ways to color a path.

Pr[Path is colorful] = $k!/k^k \geq e^{-k}$.

Pr[OPT is colorful] $\geq e^{-k}$.
Pr[OPT is not colorful] $< (1-e^{-k})$

What if the optimal path is not colorful?

Have to repeat this procedure enough times so that the probability that that happens is low.

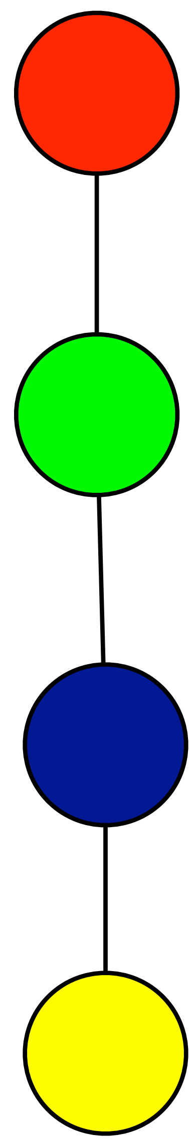
k! ways to make a path colorful.

k^k ways to color a path.

Pr[Path is colorful] = $k!/k^k \geq e^{-k}$.

Pr[OPT is colorful] $\geq e^{-k}$.

Pr[OPT is not colorful] < $(1-e^{-k})$



Repeat algorithm

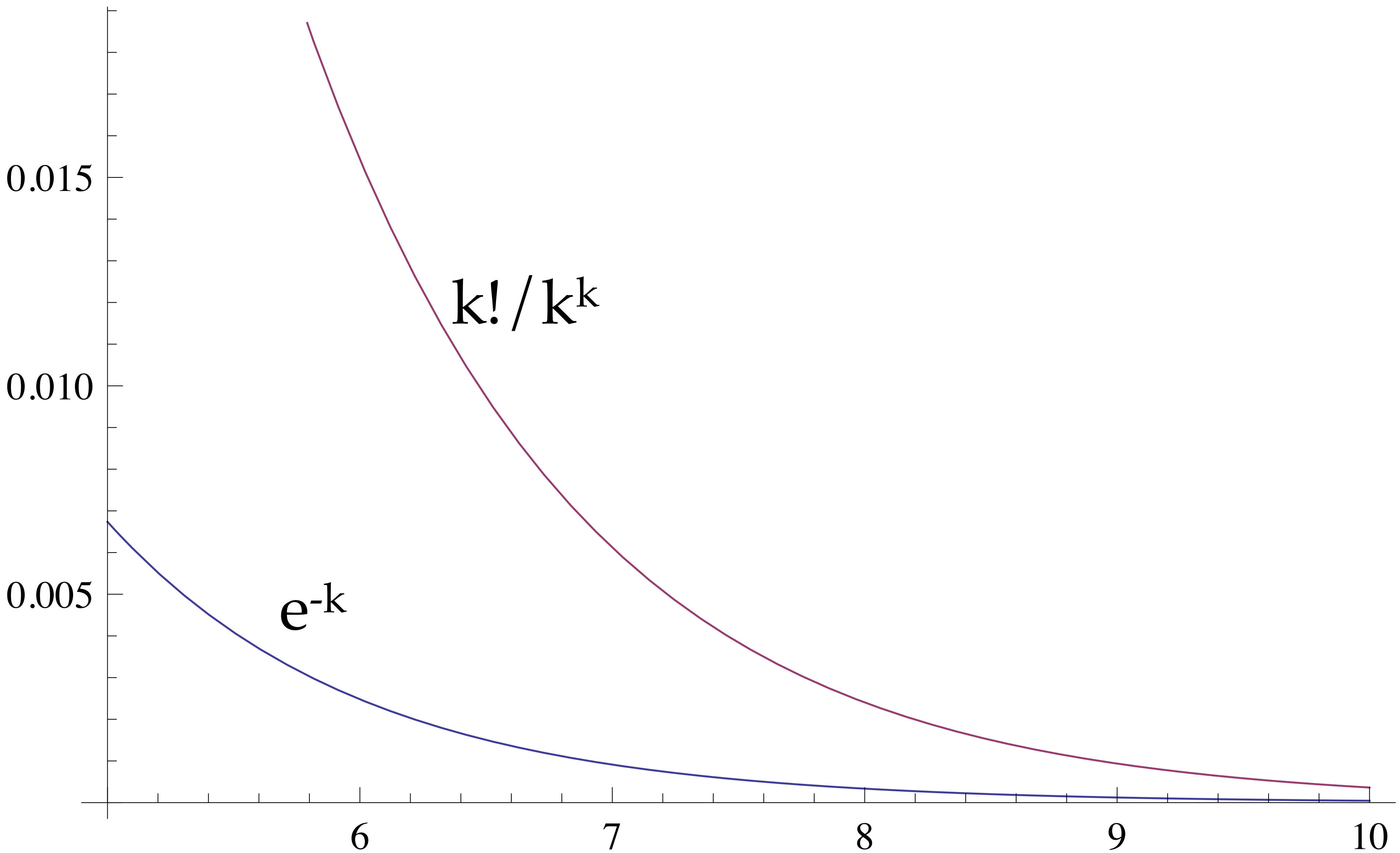
$$-e^k \ln \epsilon$$

times.

$$\Pr[\text{OPT is never colorful}] \leq$$

$$\left(1 - e^{-k}\right)^{-e^k \ln \epsilon} = \left[\left(1 + \frac{1}{-e^k}\right)^{-e^k}\right]^{\ln \epsilon}$$

$$\leq e^{\ln \epsilon} = \epsilon$$

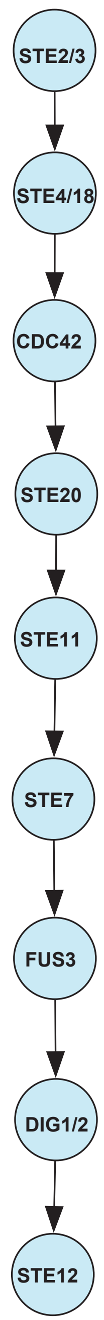

k!/k^k
e^-k
0.015
0.010
0.005
6
7
8
9
10

Running Times

Yeast Network with ~4,500 nodes and ~14,500 edges:

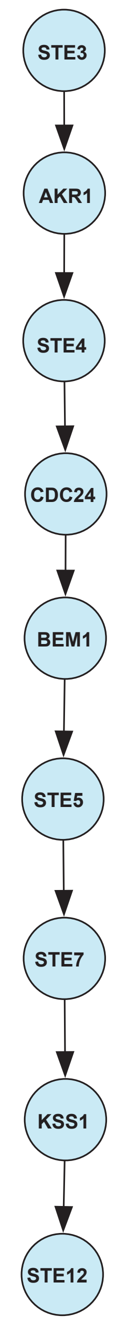
Path length	Success probability	#Paths	Time (sec)
10	99.9%	100	5613
9	99.9%	100	1241
8	99.9%	500	322
8	99.9%	300	297
8	99.9%	100	294
8	90%	100	99
8	80%	100	75
8	70%	100	61
8	50%	100	42
7	99.9%	100	86
6	99.9%	100	36

Pheromone Response Pathway



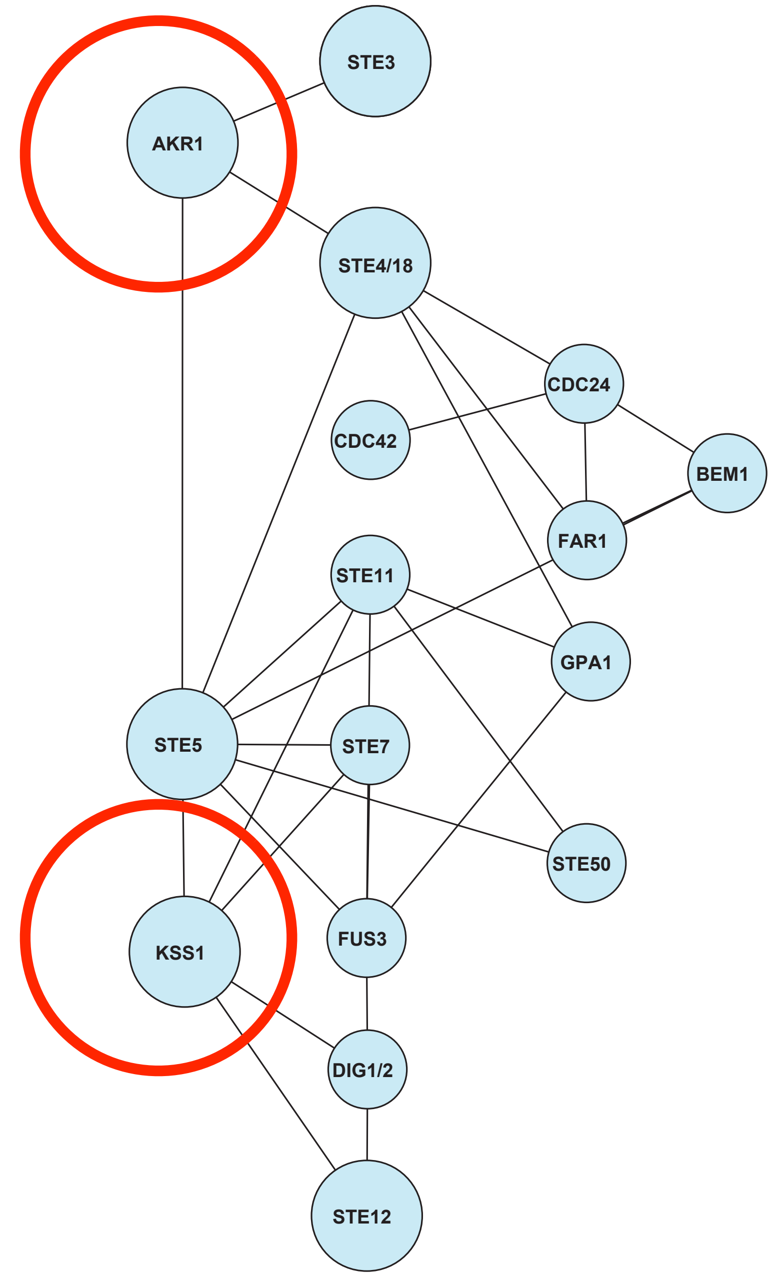
(a)

Known pathway



(b)

Best length-9 pathway between STE3 and STE12



(c)

Collection of all low-weight paths between STE3 and STE12

Color Coding Summary

- Turned a slow, $O(n^k)$ algorithm into a less-slow $O(2^k)$ algorithm that is correct with high probability.
- Used on yeast to identify signaling pathways.
- Directly extends to finding good-scoring pathways in the alignment graph of PathBLAST.

- Color Coding: Alon et al, 1995.