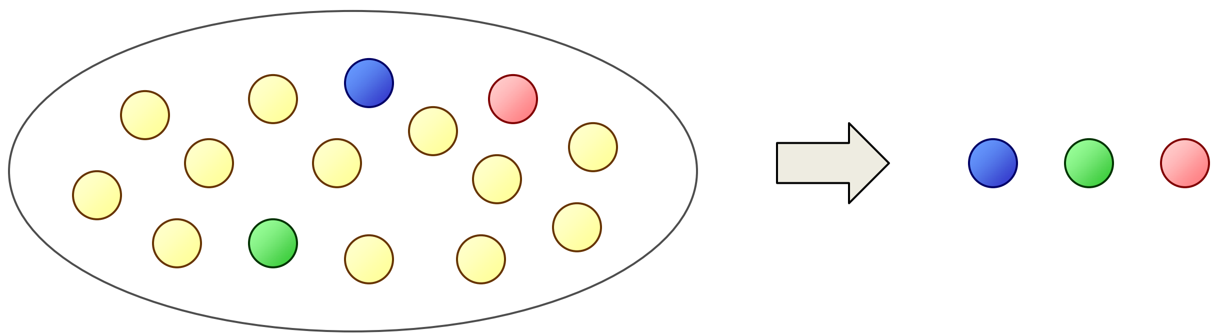
Randomized Decoding for Selection-and-Ordering Problems

Pawan Deshpande, Regina Barzilay, David R. Karger
MIT

Selection-and-Ordering Problems in NLP

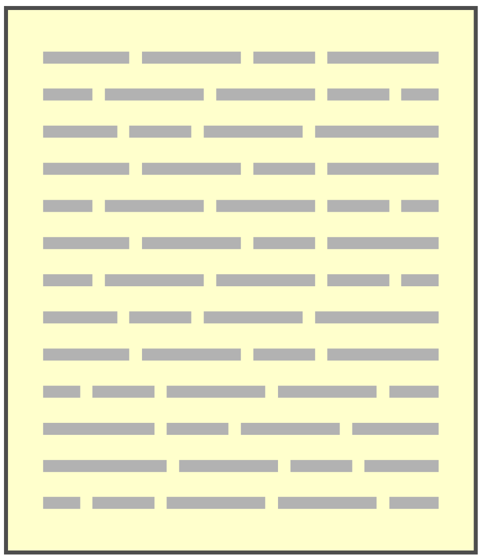
- Task: Select k units and order them optimizing some scoring function



- We assume each unit has a selection score
- We assume each pair of units has a pairwise ordering score

- Selection-and-Ordering problems appear in multiple NLP applications
  - Title generation
  - Multi-document summarization
  - …

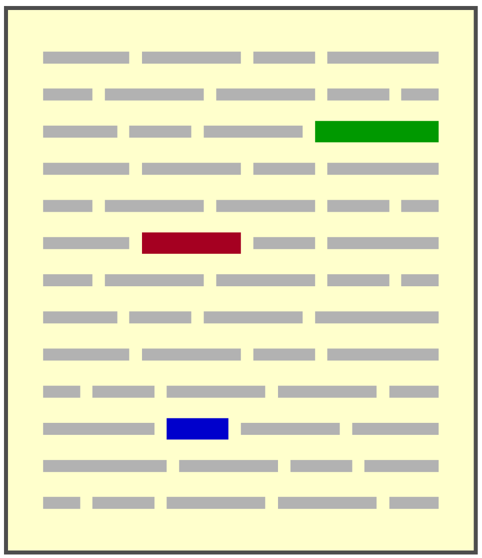
Selection-and-Ordering for Title Generation



Document Text

Input : **Document Text**
Selection Units : **Words**
Output : **Generated Title**

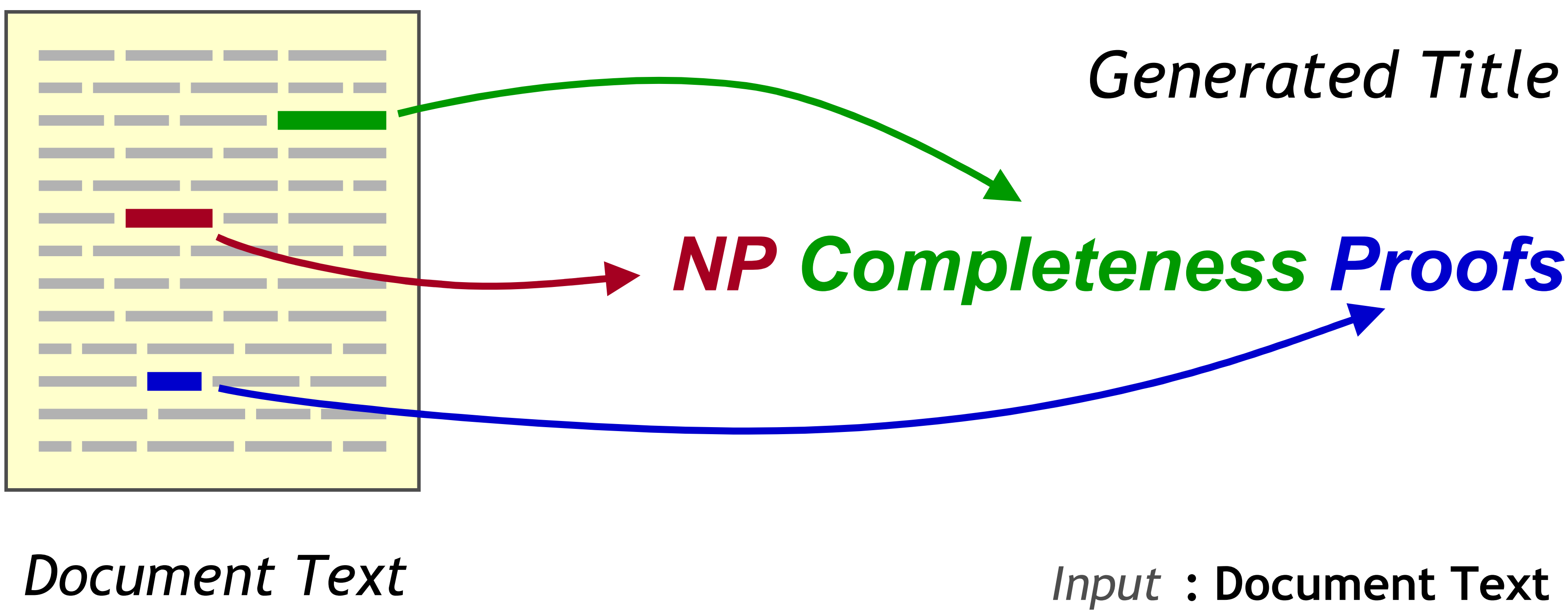
Selection-and-Ordering for Title Generation



Document Text

Input : **Document Text**
Selection Units : **Words**
Output : **Generated Title**

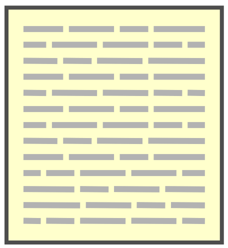
Selection-and-Ordering for Title Generation



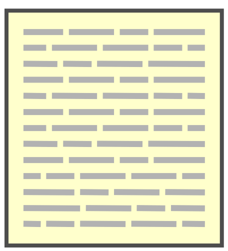
Input : **Document Text**
Selection Units : **Words**
Output : **Generated Title**

Selection-and-Ordering for Multi Document Summarization

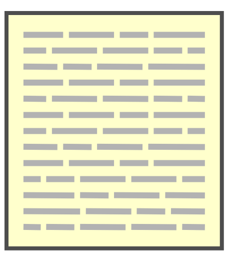
Document 1



Document 2



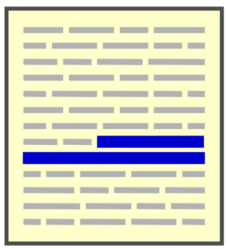
Document 3



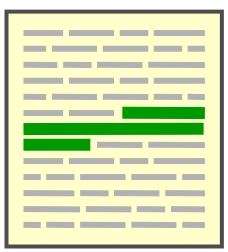
Input : **Multiple Documents**
Selection Units : **Sentences**
Output : **Generated Summary**

Selection-and-Ordering for Multi Document Summarization

Document 1



Document 2



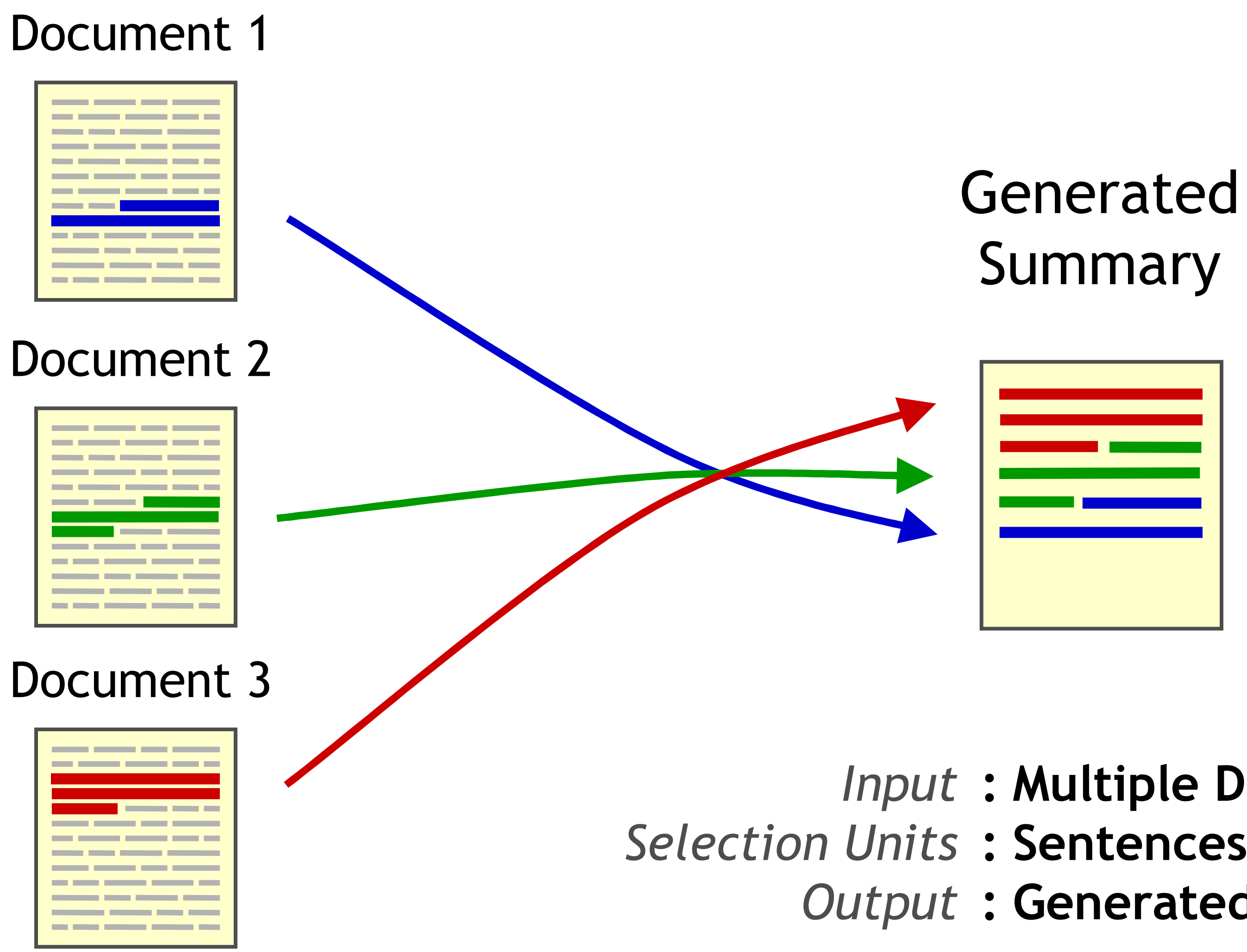
Document 3

Input : **Multiple Documents**

Selection Units : **Sentences**

Output : **Generated Summary**

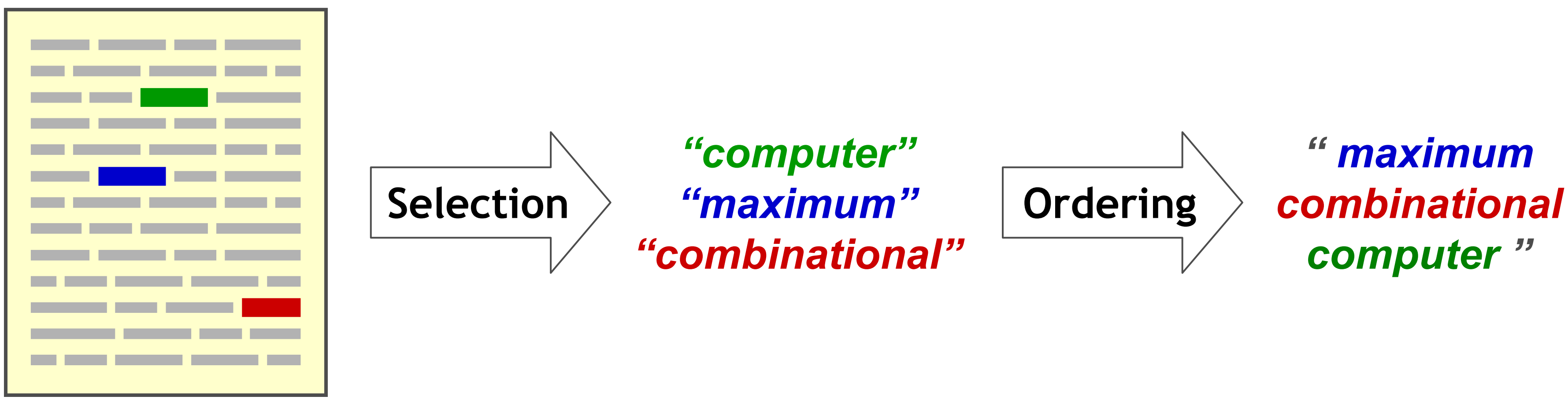
Selection-and-Ordering for Multi Document Summarization



Input : **Multiple Documents**
Selection Units : **Sentences**
Output : **Generated Summary**

Obvious Approach: Pipelining

Problem: Selected words cannot be ordered as a coherent phrase



Previous Work on Decoding

- *Inexact Decoding*
  - Beam Search
    (Banko et. al. 2000, Corston-Oliver et al. 2000, Jin and Hauptmann, 2001)
  - Sampling
    (Eisner and Tromble, 2006, Finkel et. al, 2005)

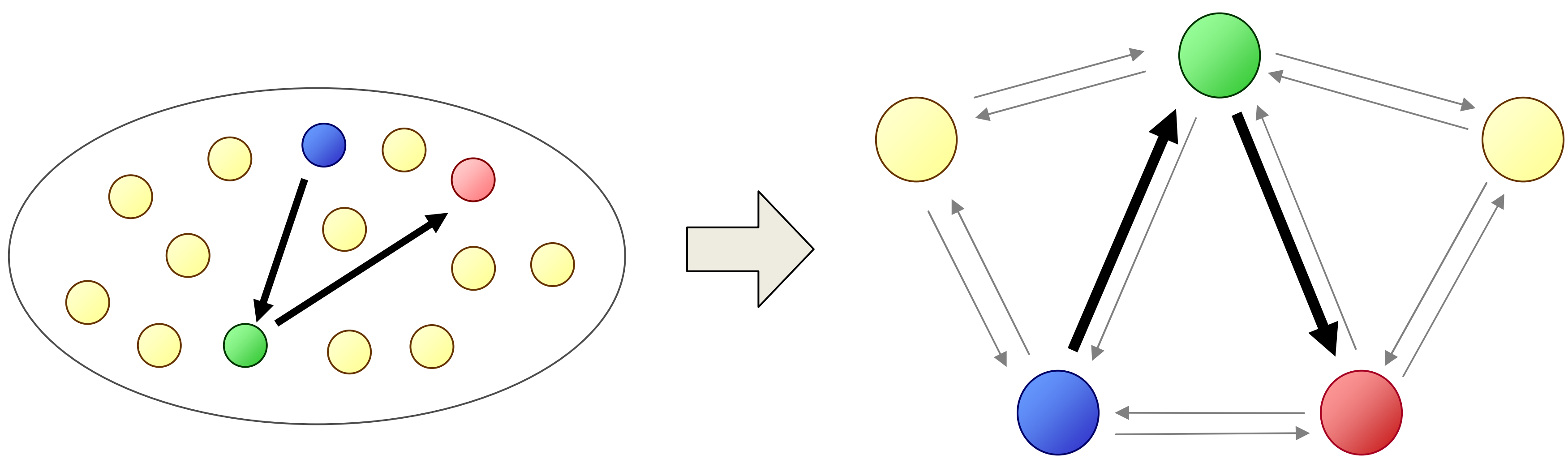
  Fast but optimal solution not guaranteed

- *Exact Decoding*
  - A*
    (Jelinek, 1969; Germann et al., 2001)
  - Integer Linear Programming
    (Germann et al. 2001, Roth and Yih, 2004)

  Optimal solution guaranteed but slow

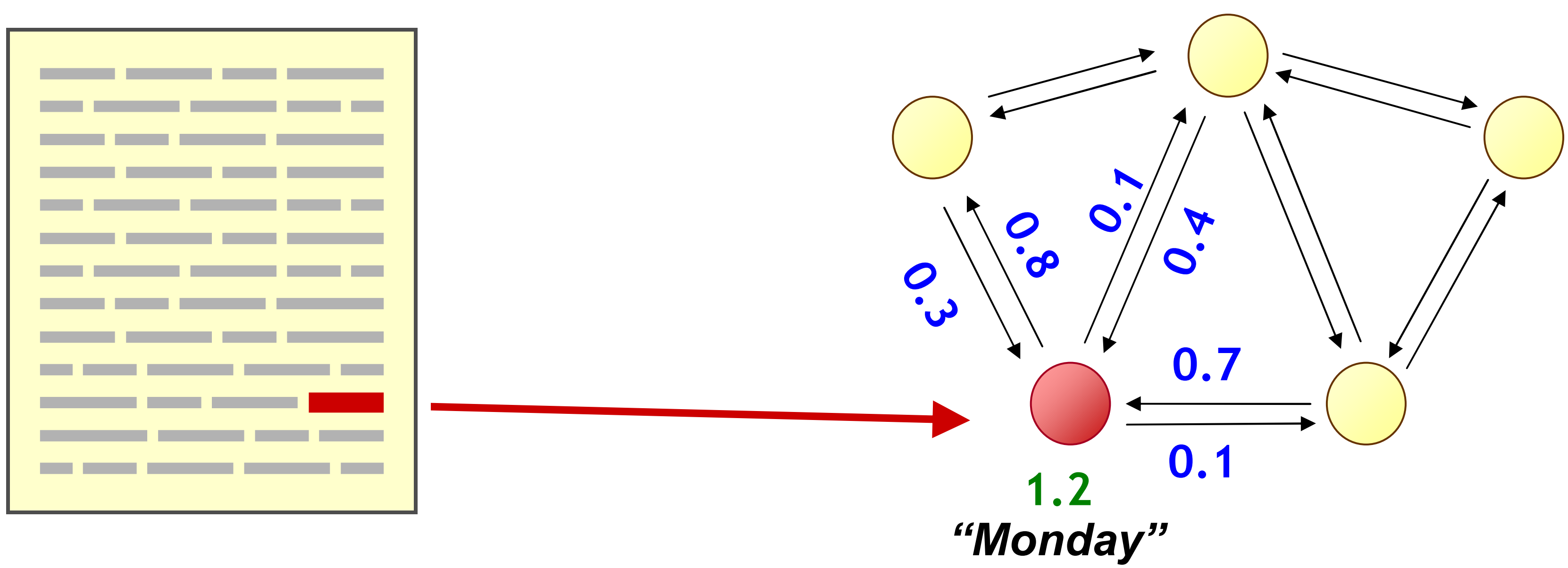
Problem Formulation

We formalize decoding for selection-and-ordering problems as the task of finding an acyclic path in a graph



Example: Graph Representation for Title Generation

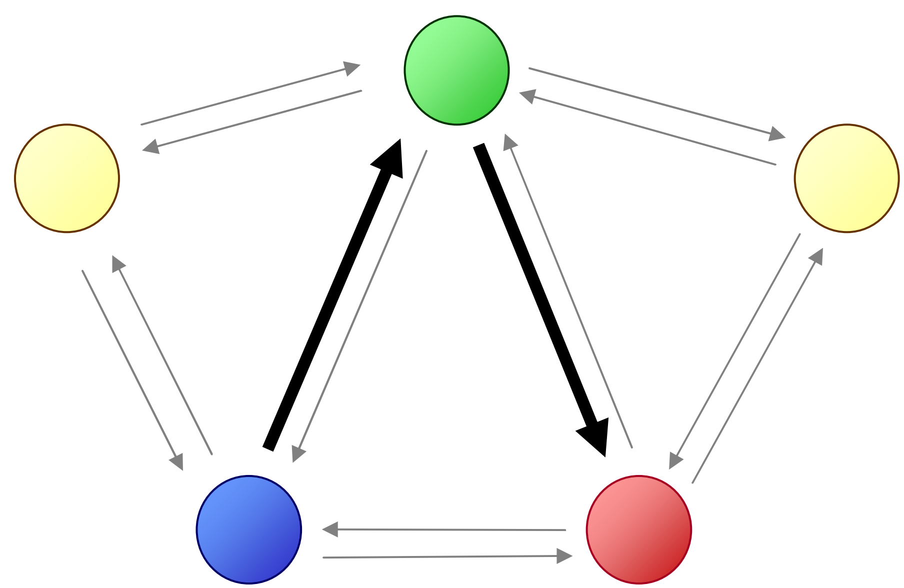
Selection Score : Likelihood of word appearing in the title
Ordering Scores : Bigram language model



Decoding Task: Title Generation

Objective: Generate a k word title that optimizes selection and ordering scores without repetition

Find the highest weighted acyclic path of length k



Decoding Complexity

- Decoding for selection-and-ordering is NP-hard
  - Instance of *prize collecting traveling salesman problem* (Balas, 1989; Awerbuch et al., 1995)
- The proposed randomized method has runtime of $O(2^k n^2)$
  - n is the total number of units
  - k is the number of selected units
  - $k << n$

Guarantees an arbitrary bound on the likelihood of finding the optimal solution

Decoding with Color-coding

Our approach: graph theoretic randomized algorithm

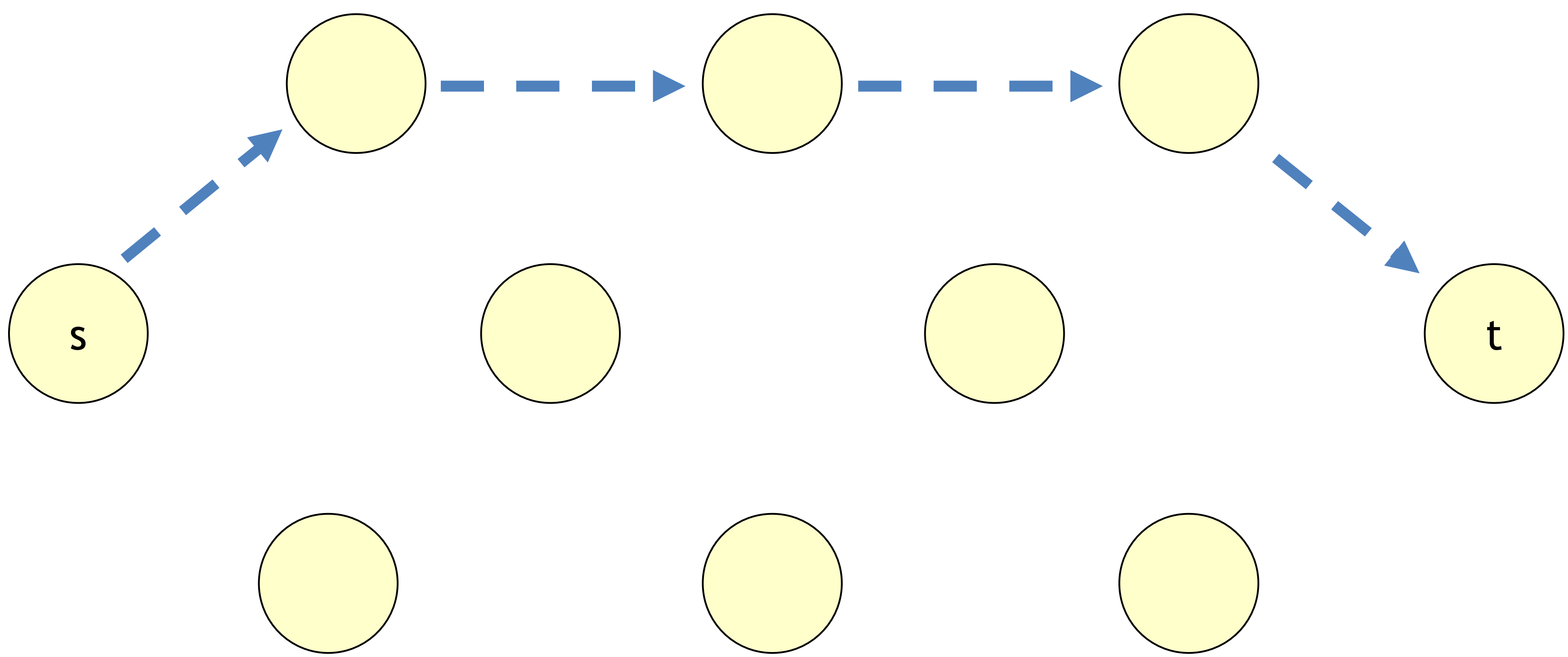
- Randomly color vertices
- Find highest weighted path with no repeated colors
- Repeat

Color-coding belongs to the family of algorithms theoretically analyzed by Alon et. al (1995)

Step 1: Randomly Color Vertices

- Randomly color vertices with r colors

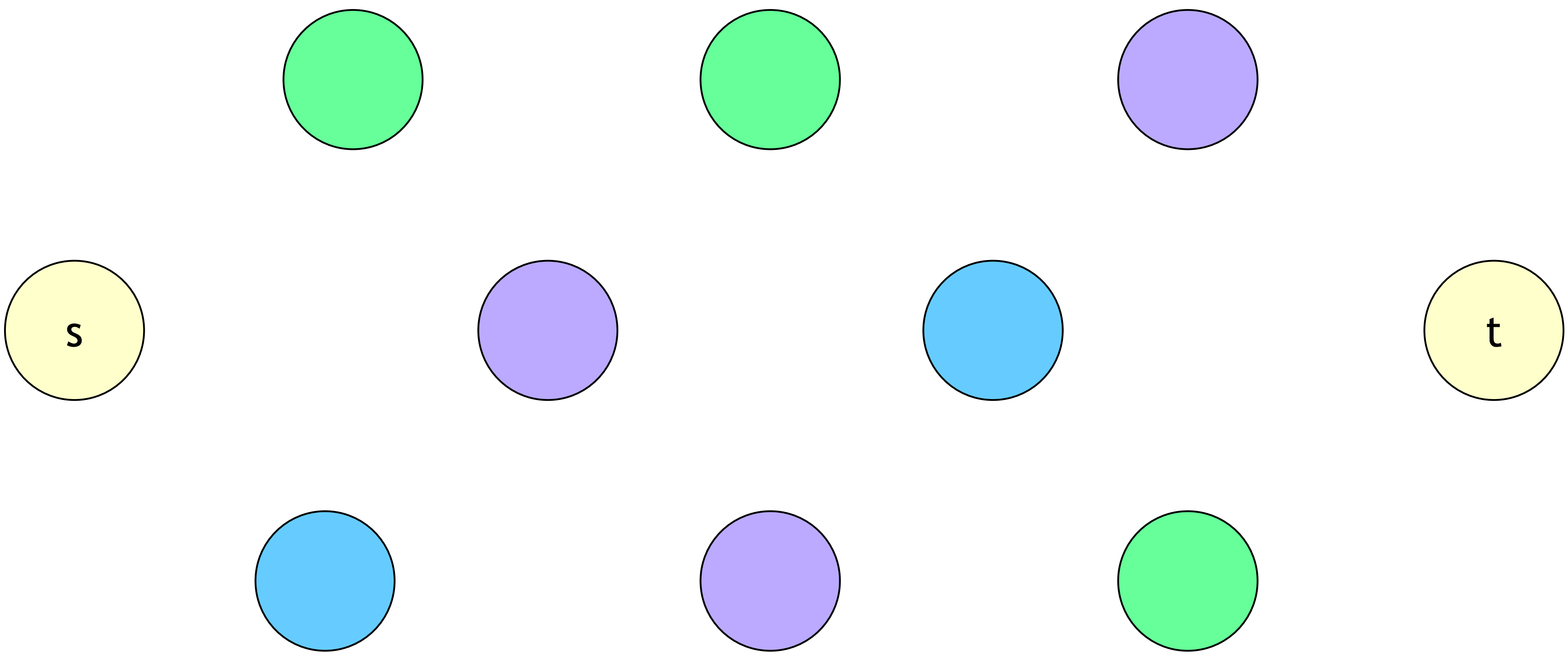
Number of colors $r = 3$.   Selected Units $k = 3$



Step 1: Randomly Color Vertices

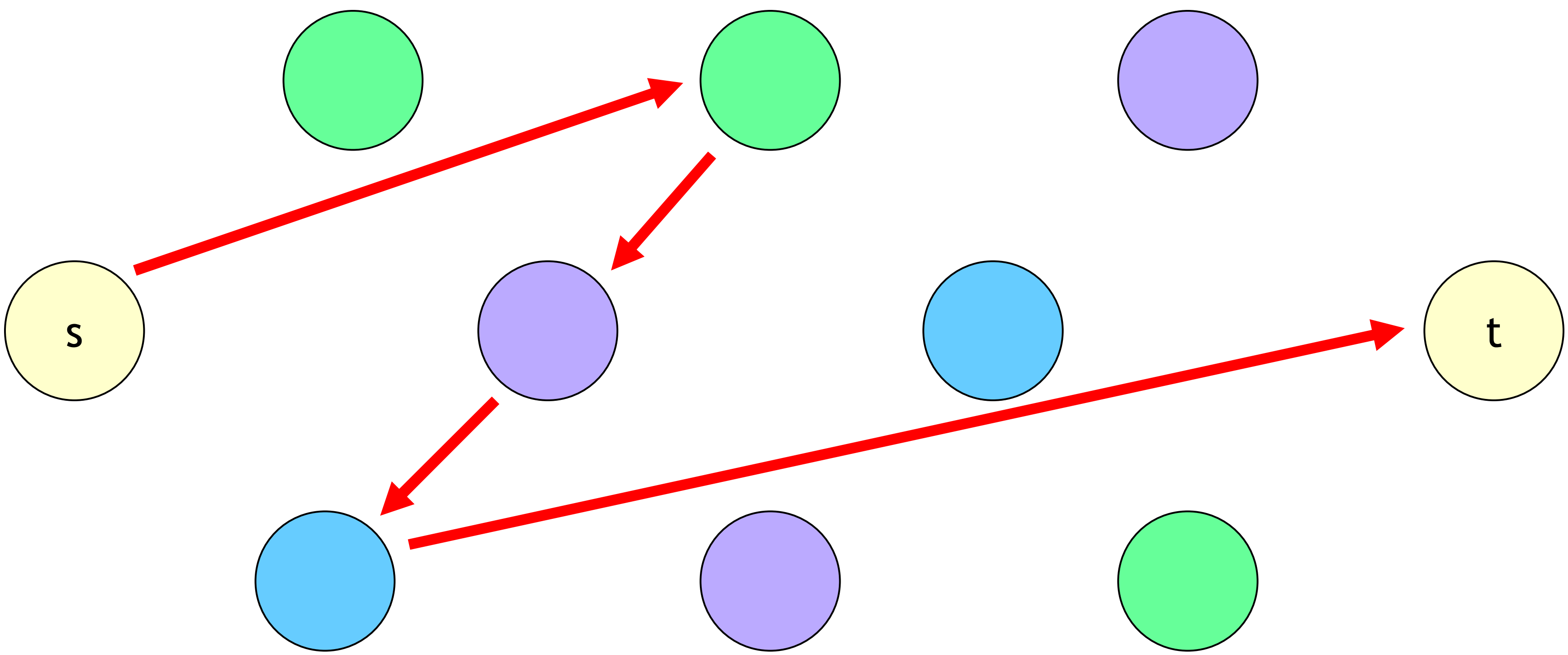
- Randomly color vertices with r colors

Number of colors $r = 3$.   Selected Units $k = 3$



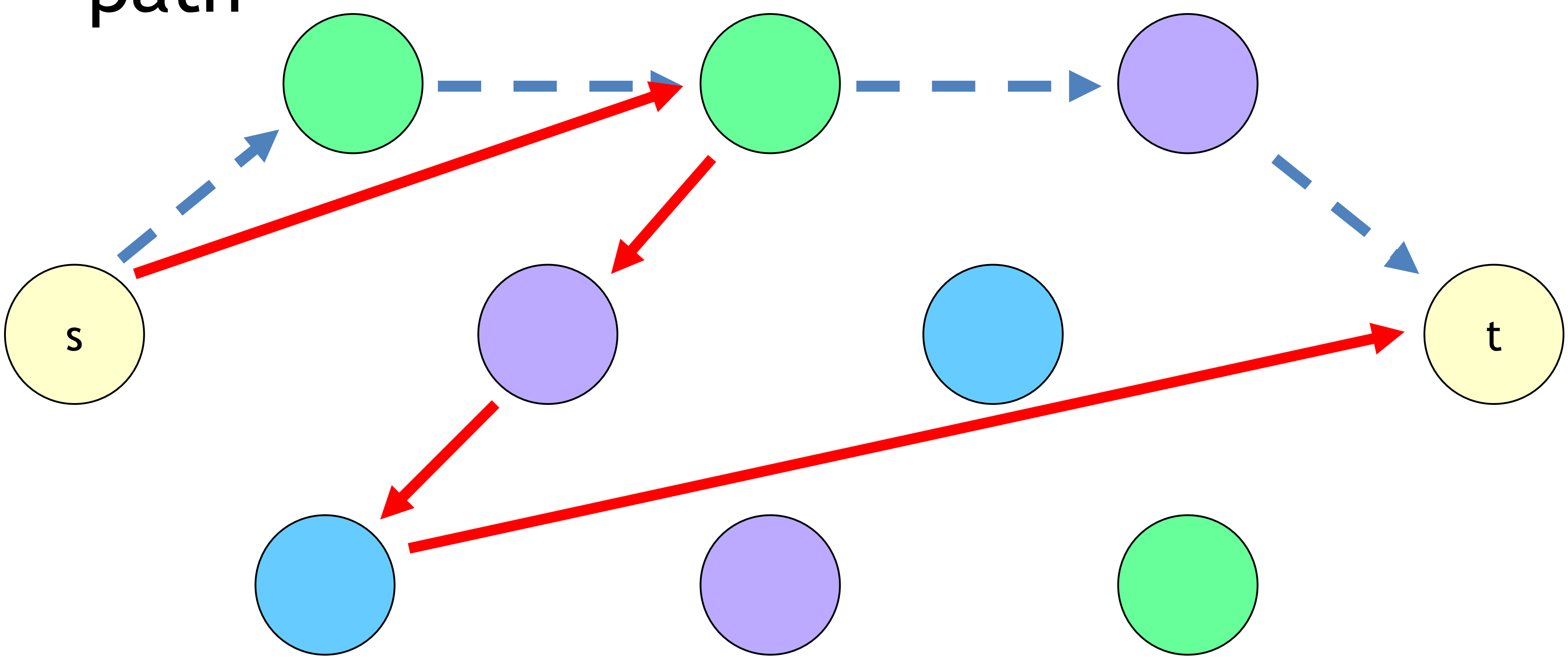
Step 2: Find highest weighted k-colorful path

- Find the highest weighted path of length k with no repeated colors



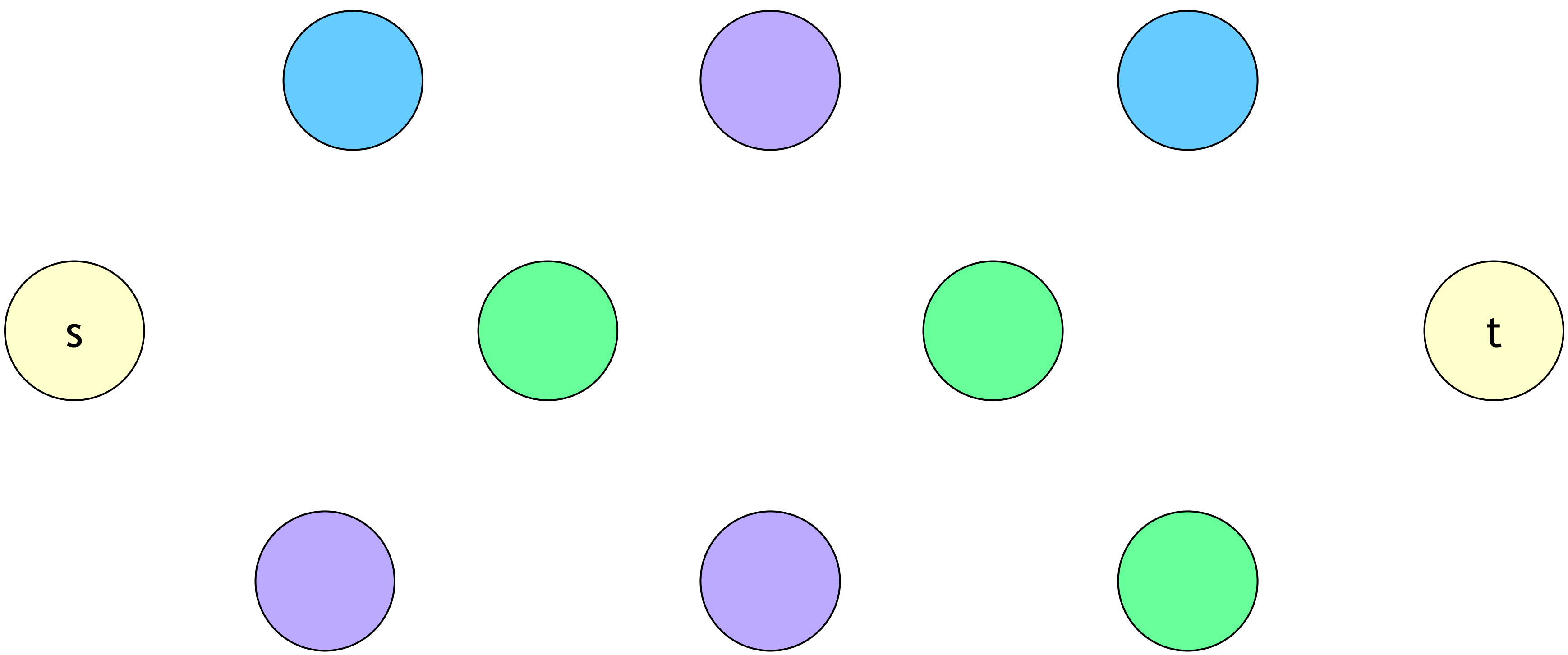
Step 3: Repeat

- Single run may not return the optimal path
- Repeat to increase probability of optimal path



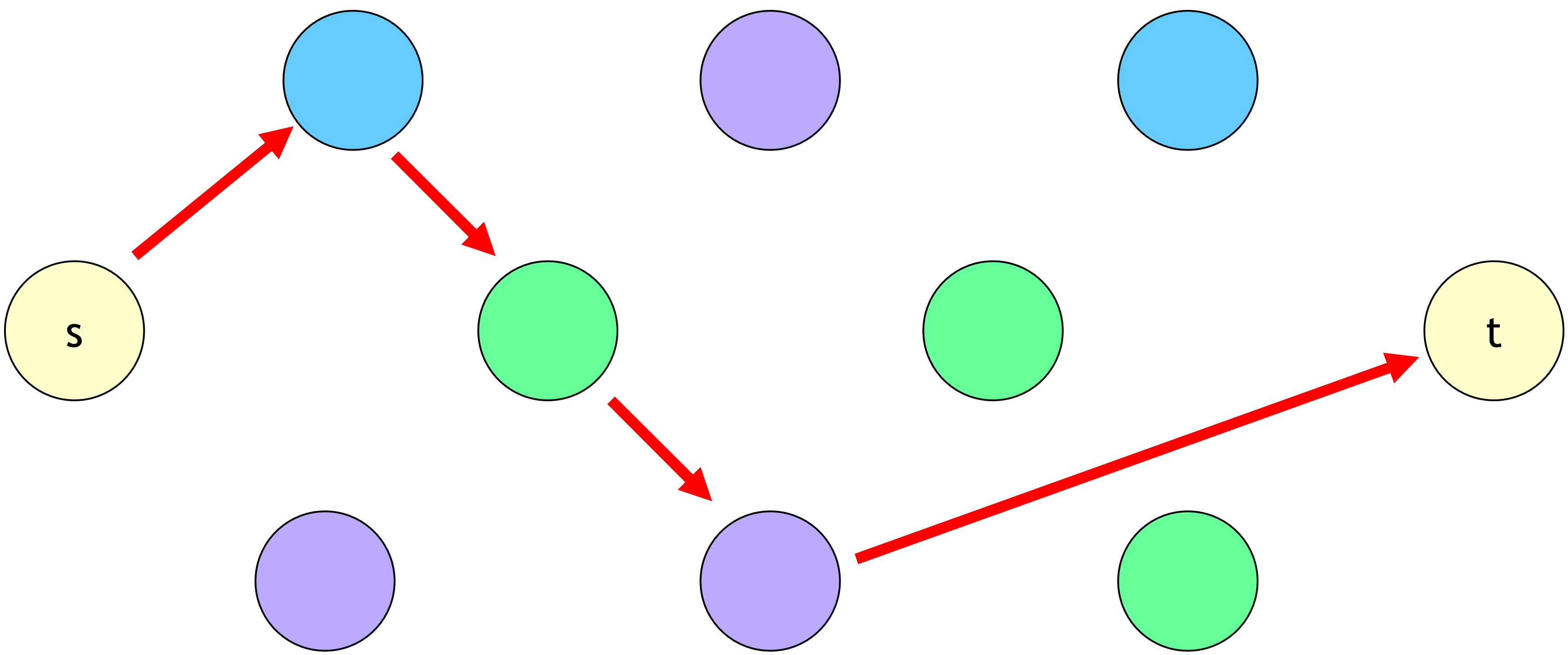
Step 3: Repeat

- Repeat and maintain the best path



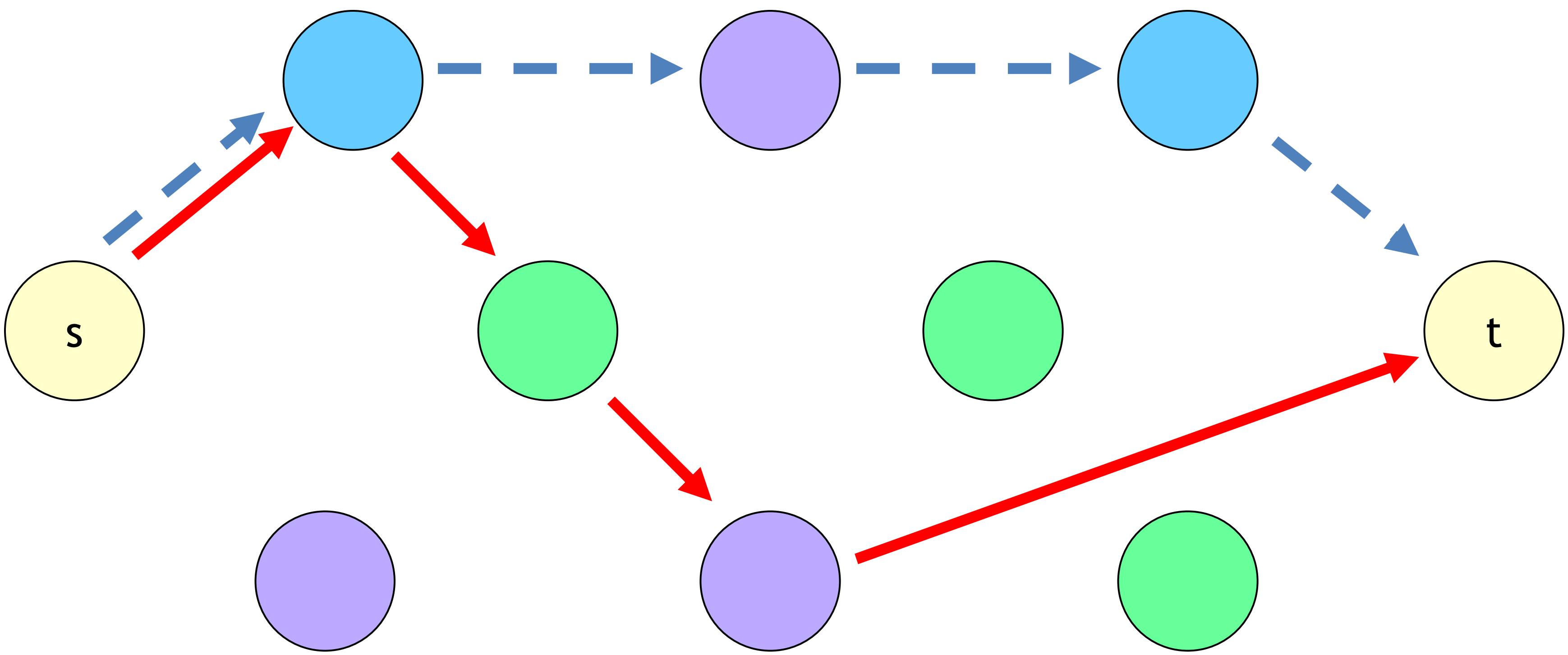
Step 3: Repeat

- Repeat and maintain the best path

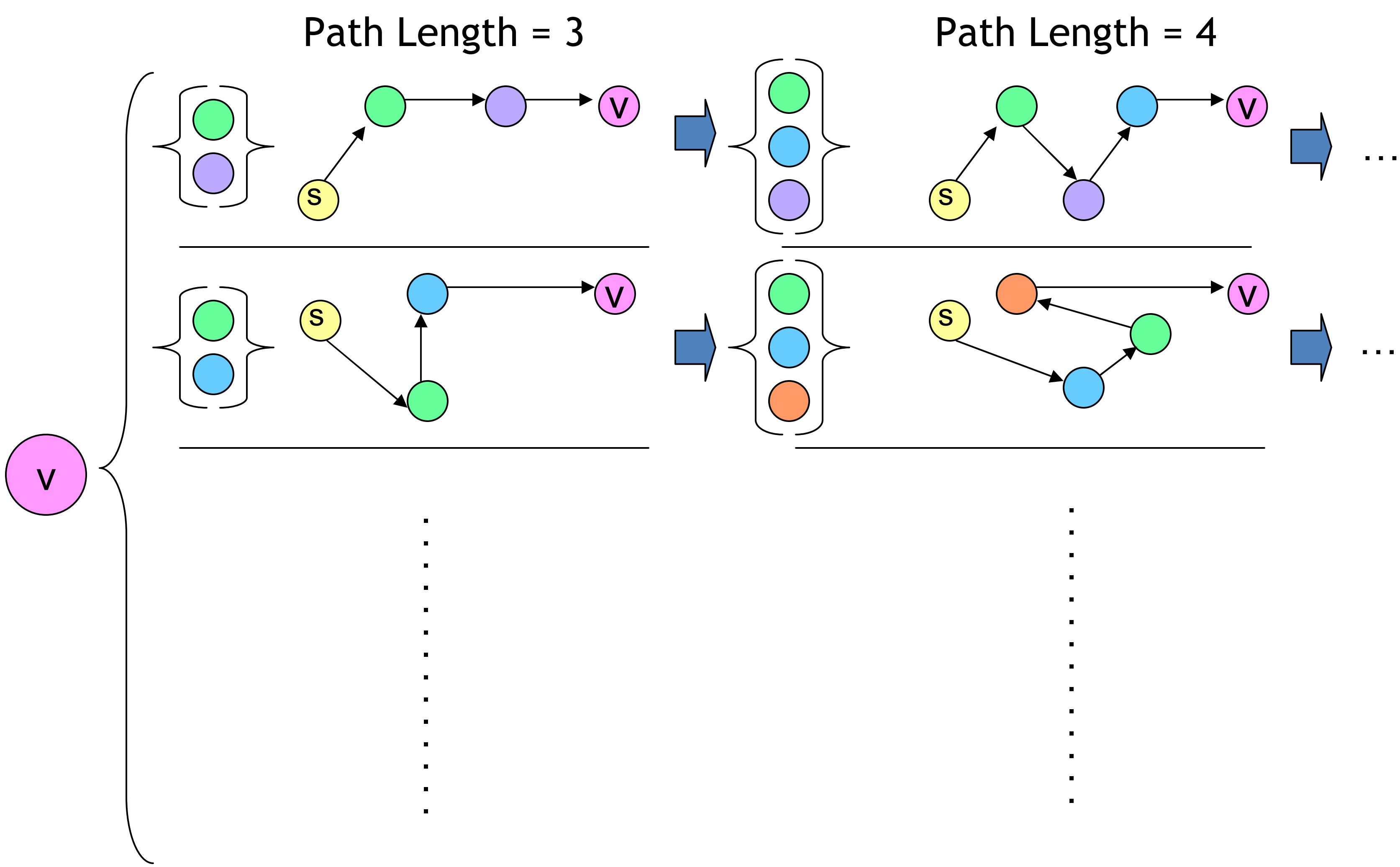


Step 3: Repeat

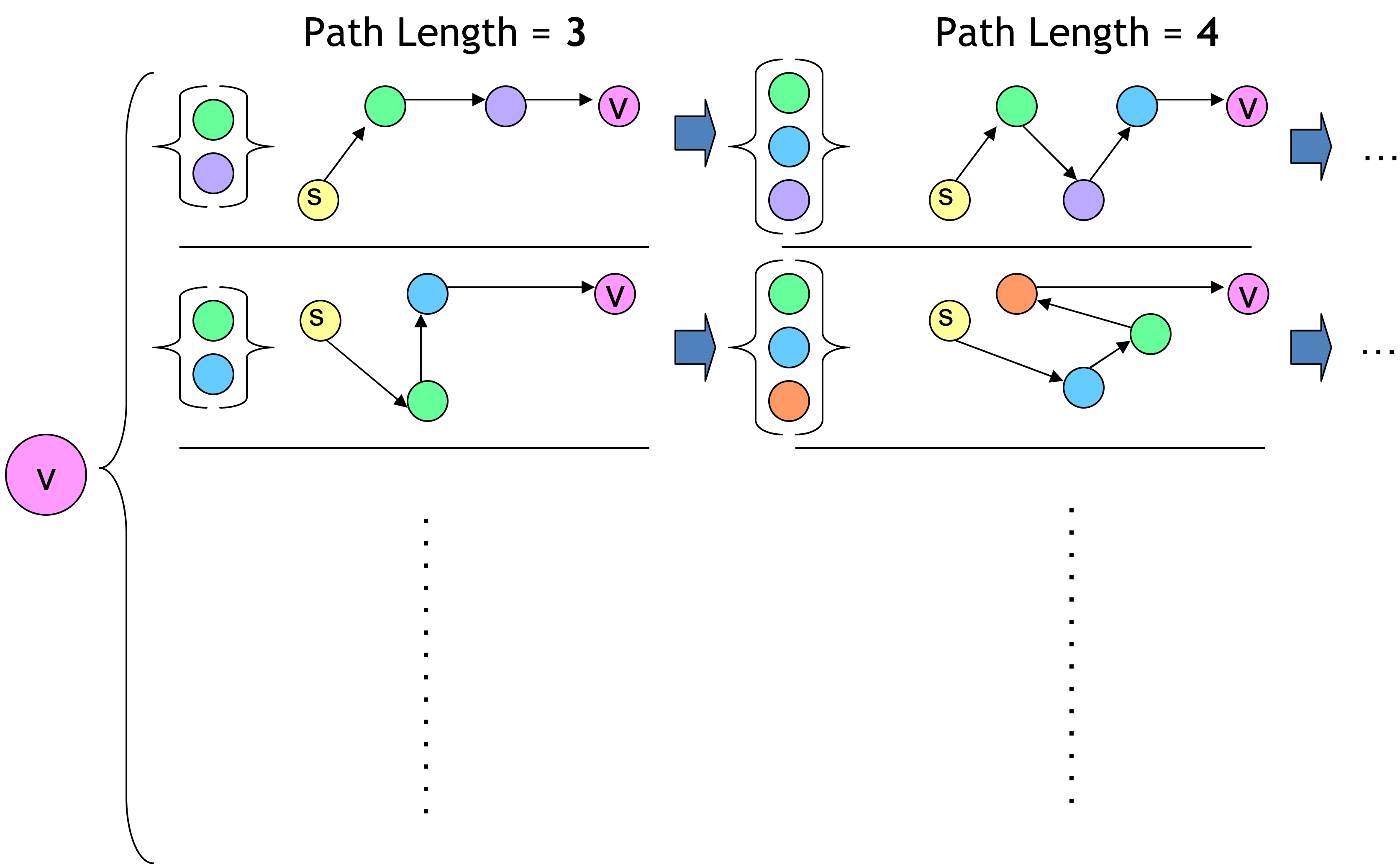
- Repeat and maintain the best path



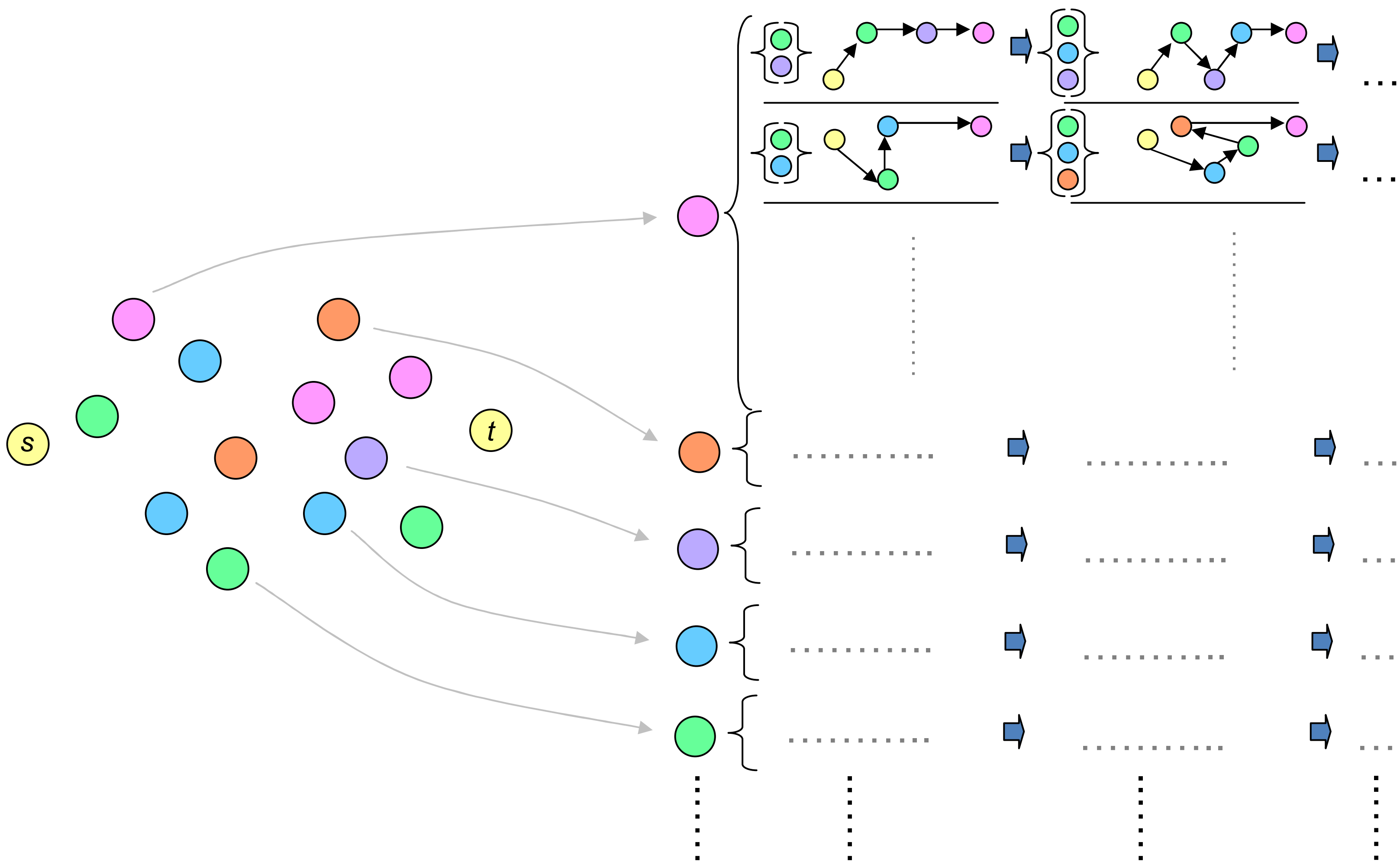
Dynamic Programming



Dynamic Programming

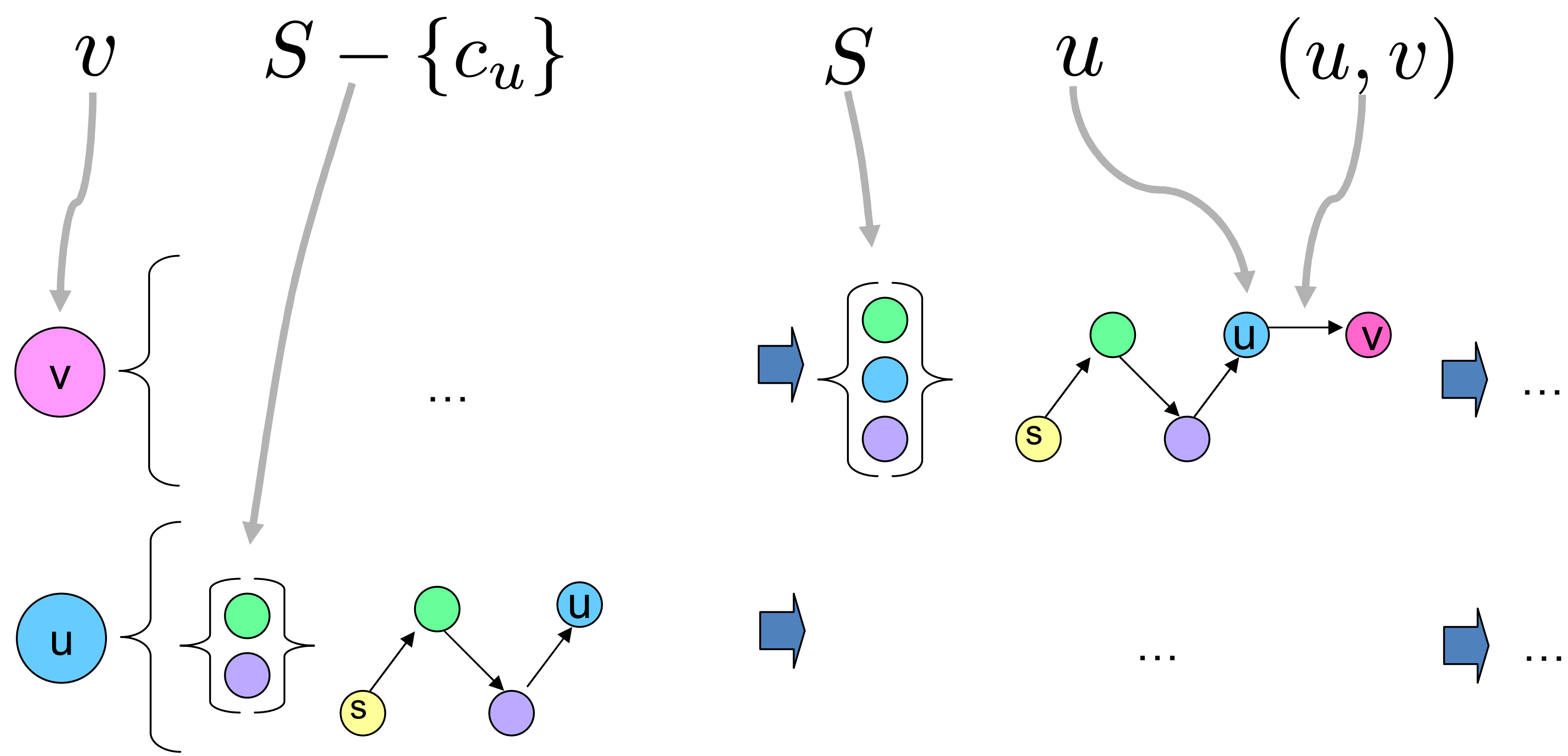


Dynamic Programming



Dynamic Programming

$$q[v, S] = \max_{(u,v)\in G, c_u \in S, c_v \notin S} q[u, S - \{c_u\}] + w(u, v)$$



Theoretical Guarantees

- Desired probability of overall success : $\mathbf{1}$- $\boldsymbol{\delta}$
- Single run probability of success : $\boldsymbol{p}$
  - In our case $p = 1 \cdot \frac{r-1}{r} \cdot \frac{r-2}{r} \cdots \frac{r-(k-1)}{r}$
- Success on a single run yields overall success
- Probability of failing on all $\boldsymbol{t}$ runs :

$$(1-p)^t \le e^{-pt} = \delta$$

- Need to run algorithm $\mathbf{(1/\boldsymbol{p})\ ln(1/\boldsymbol{\delta})}$ times

Finding the number of colors

- Probability that the optimal path has no repeated colors: $1 \cdot \frac{r-1}{r} \cdot \frac{r-2}{r} \cdots \frac{r-(k-1)}{r}$
- Running time for required number of trials T_r will be proportional to

$$\overbrace{1 \cdot \frac{r}{r-1} \cdot \frac{r}{r-2} \cdots \frac{r}{r-(k-1)}}^{\textit{Required number of trials}} \cdot \overbrace{2^r}^{\textit{Running time per trial}}$$

- Solve for r where T_r is minimized

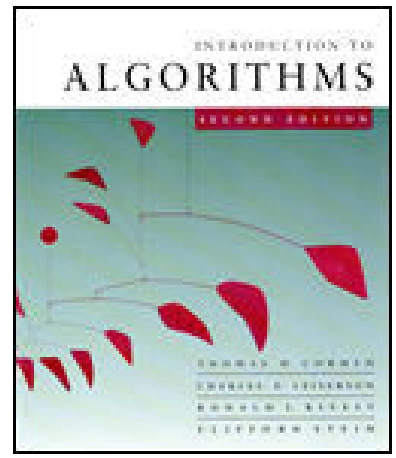
  $r \approx 1.302\, k$

Experimental Set-Up

- **Task:** Title Generation
- **Method:** Based on Banko et. al (2000)
- **Selection scores:** Max. entropy classifier
  - Distributional Features (e.g. TF-IDF)
  - Positional Features (e.g. First occurrence in text)
- **Ordering scores:** Bigram language model
  - Computed from words in document text
  - Computed from part-of-speech tags of document titles

Corpus

- **Corpus:** 547 sections of algorithms textbook
  - Training set: 382 sections
  - Test set: 165 sections



- **Corpus Statistics**
  - Average title length: 3.7 words
  - Average section length: 609.2 words
  - Redundancy: 97.9% of titles do not contain repeated words

Decoding Algorithms

- **Color-coding**
  - Parallelized computation of coloring iterations
- **Beam Search**
  - Modified to enforce acycility
- **Integer Linear Programming**
  - Formulation is based on max-flow problem
  - Solved using Mixed Integer Programming solver *lp_solve*

Results

Algorithm	Average (s)	Median (s)	ROUGE-L	Optimal Solutions (%)
Beam 1	0.6	0.4	0.0234	0.0
Beam 80	28.4	19.3	0.2373	64.8
Beam 1345	368.6	224.4	0.2556	100.0
ILP	6,536.2	57.3	0.2556	100.0
Color-coding	73.8	9.7	0.2556	100.0

- ILP is slow – Up to 136 hours!
- Large beam required for finding all optimal solutions
- Color-coding can find the optimal solutions quickly
- Stronger decoding strategies improve title quality

Results

Algorithm	Average (s)	Median (s)	ROUGE-L	Optimal Solutions (%)
Beam 1	0.6	0.4	0.0234	0.0
Beam 80	28.4	19.3	0.2373	64.8
Beam 1345	368.6	224.4	0.2556	100.0
ILP	***6,536.2***	***57.3***	***0.2556***	***100.0***
Color-coding	73.8	9.7	0.2556	100.0

- **ILP is slow - Up to 136 hours!**
- Large beam required for finding all optimal solutions
- Color-coding can find the optimal solutions quickly
- Stronger decoding strategies improve title quality

Results

Algorithm	Average (s)	Median (s)	ROUGE-L	Optimal Solutions (%)
Beam 1	0.6	0.4	0.0234	0.0
Beam 80	28.4	19.3	0.2373	64.8
Beam 1345	***368.6***	***224.4***	***0.2556***	***100.0***
ILP	6,536.2	57.3	0.2556	100.0
Color-coding	73.8	9.7	0.2556	100.0

- ILP is slow – Up to 136 hours!
- **Large beam required for finding all optimal solutions**
- Color-coding can find the optimal solutions quickly
- Stronger decoding strategies improve title quality

Results

Algorithm	Average (s)	Median (s)	ROUGE-L	Optimal Solutions (%)
Beam 1	0.6	0.4	0.0234	0.0
Beam 80	28.4	19.3	0.2373	64.8
Beam 1345	368.6	224.4	0.2556	100.0
ILP	6,536.2	57.3	0.2556	100.0
Color-coding	***73.8***	***9.7***	***0.2556***	***100.0***

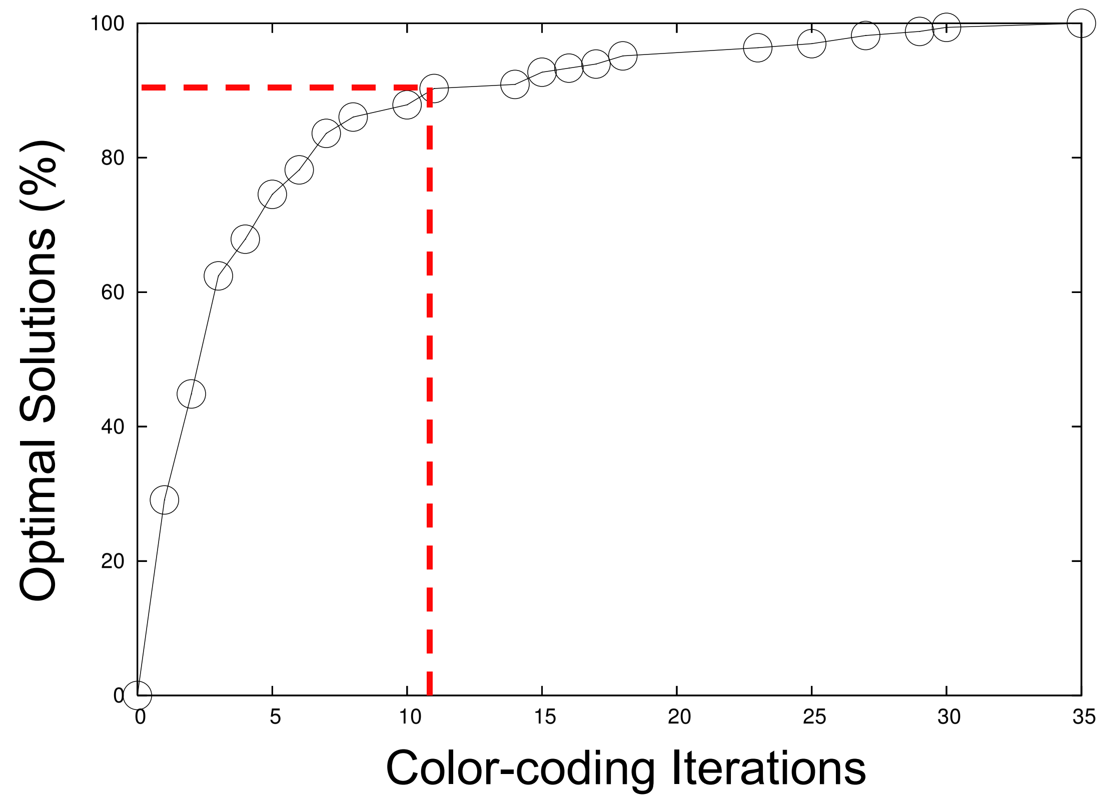
- ILP is slow – Up to 136 hours!
- Large beam required for finding all optimal solutions
- ***Color-coding can find the optimal solutions quickly***
- Stronger decoding strategies improve title quality

Results

Algorithm	Average (s)	Median (s)	ROUGE-L	Optimal Solutions (%)
Beam 1	0.6	0.4	***0.0234***	***0.0***
Beam 80	28.4	19.3	***0.2373***	***64.8***
Beam 1345	368.6	224.4	***0.2556***	***100.0***
ILP	6,536.2	57.3	***0.2556***	***100.0***
Color-coding	73.8	9.7	***0.2556***	***100.0***

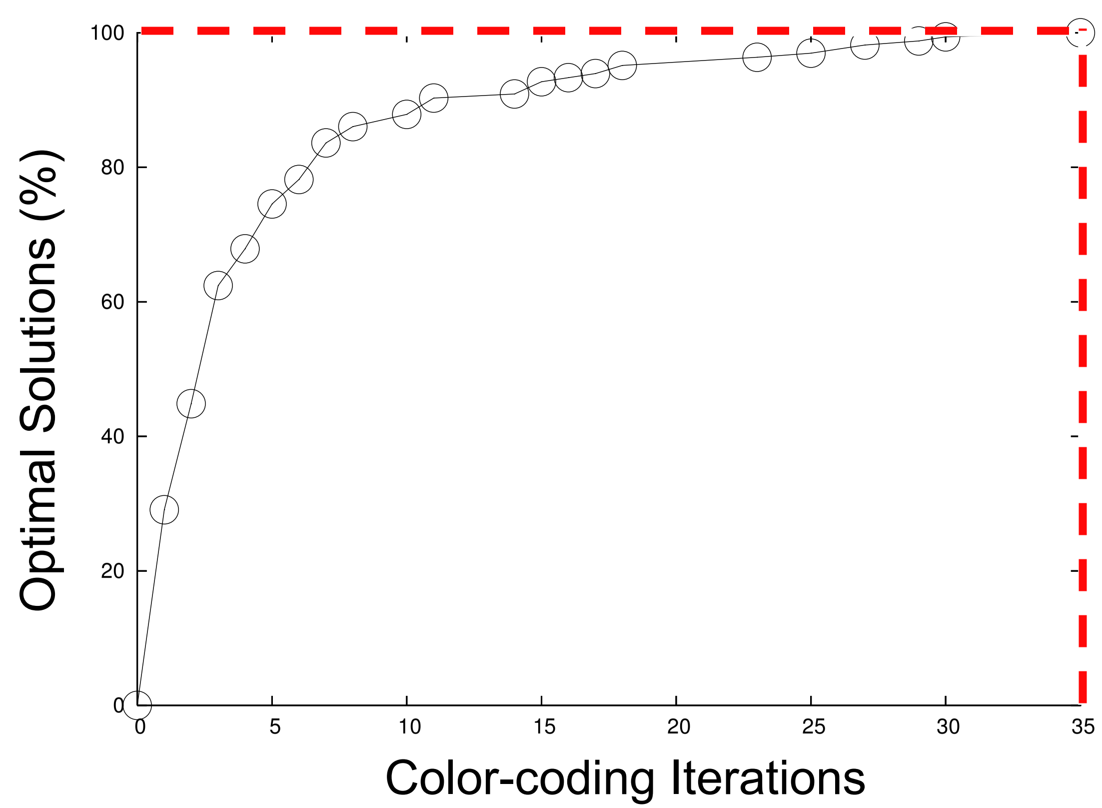
- ILP is slow – Up to 136 hours!
- Large beam required for finding all optimal solutions
- Color-coding can find the optimal solutions quickly
- **Stronger decoding strategies improve title quality**

Color-coding Convergence



- 11 iterations: 90% of solutions found
- 35 iterations: All optimal solutions

Color-coding Convergence



- 11 iterations: 90% of solutions found
- 35 iterations: All optimal solutions

Conclusions

- Color-coding offers theoretical guarantees on its performance and time-complexity
- In practice, it outperforms existing decoding methods
- Randomized algorithms open doors to a new class of decoding techniques

Future Work

- Explore additional constraints within the color-coding framework
- Apply color-coding to other NLP tasks

Code for algorithms

`http://people.csail.mit.edu/pawand/rand/`