Graph ranking
(PageRank, SimRank)

Hyunjoong Kim

soy.lovit@gmail.com

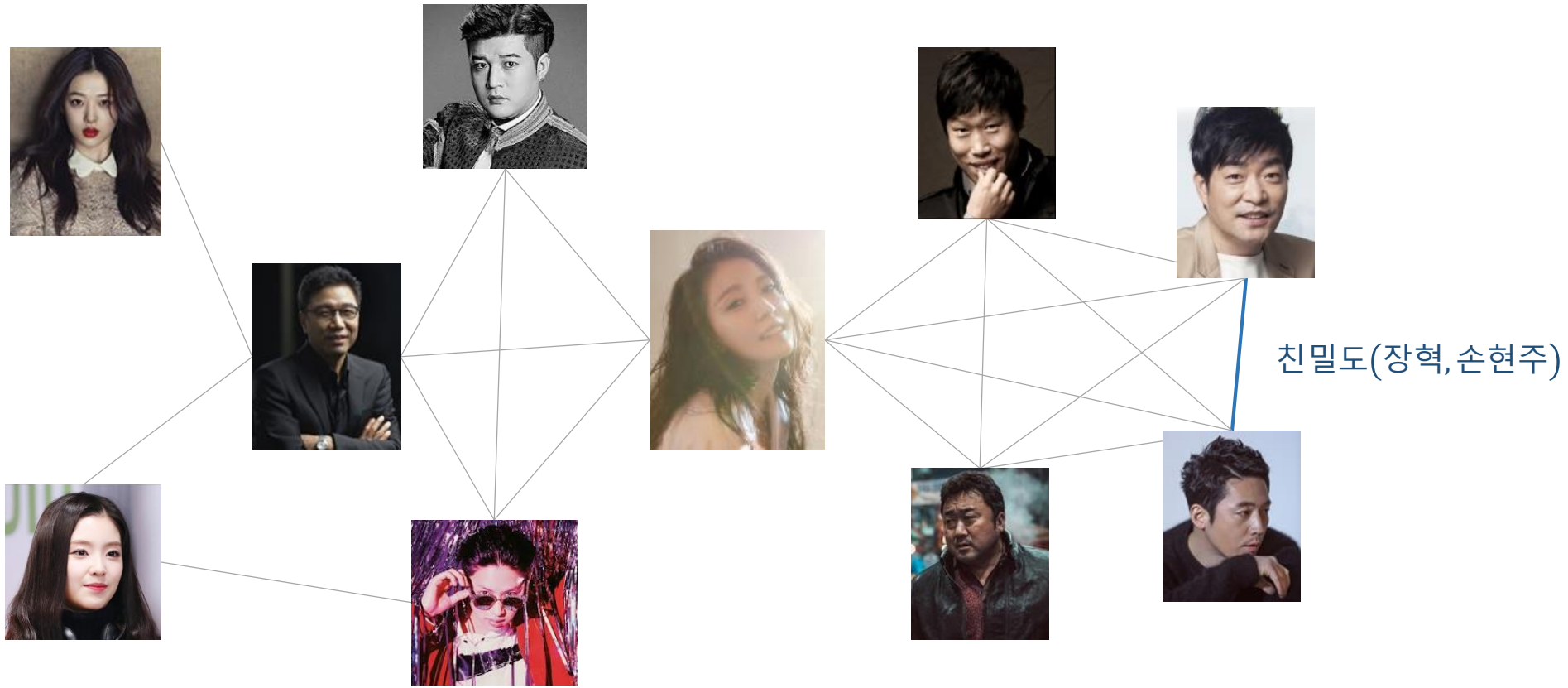
github.com/lovit/{soygraph, textrank, kr-wordrank}

Introduction

- 그래프는 데이터를 표현하는 형식입니다.
  - 그래프는 마디(node, V) 와 호(edge, E)로 이뤄져 있습니다.
  - G = (V, E)
- 벡터 공간보다 더 자유로운 표현이 가능합니다.
  - (N1, N2), (N1, N3) 는 가깝지만, (N2, N3) 이 매우 멀 수 있습니다.

Introduction

- “Social media user networks”

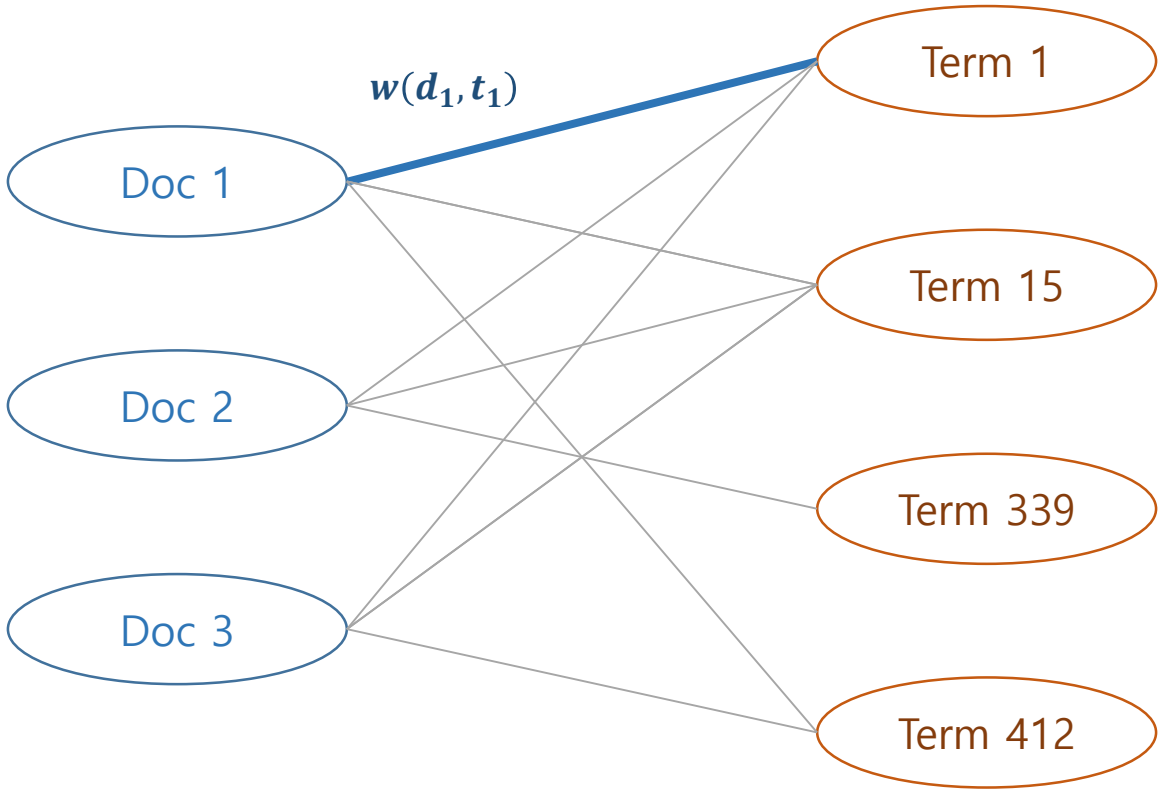


Introduction

- “영화 – 배우” 그래프

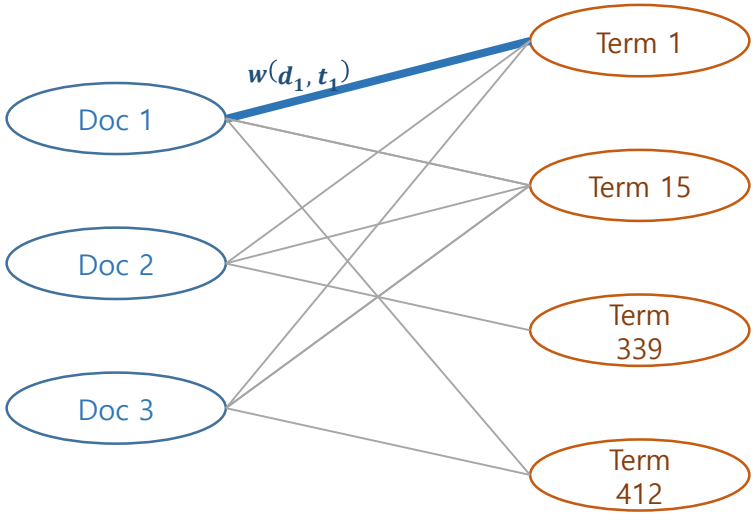
Introduction

- Term document graph



Introduction

- (row, column) 을 edge 의 시작점과 끝점으로 설정하면 그래프는 행렬로 표현됩니다.
  - Edge 는 두 마디 간의 거리 / 유사도 / 가중치 등의 값을 부여할 수 있습니다.



graph form

	1	2	...	15	...	339	412	...
D1	4	-	...	3	...	-	2	...
D2	2	-	...	6	...	4	-	...
D3	5	-	...	4	...	-	5	...
D4	-	2	...	-	...	-	3	...

matrix form

Introduction

- “영화 – 배우“나 “문서 – 단어“ 처럼 마디의 종류가 두 가지로 나뉘어지고, 서로 다른 종류의 마디끼리만 연결이 된 경우, “bipartite graph”라 합니다.

Introduction

- 그래프 형식으로 표현된 데이터에 대한 대표적인 질문은
  - Which nodes are important?
  - Which nodes are similar with given node?

Which nodes are important?

- 마디의 중요도를 정의하는 다양한 statistics 이 있습니다.
  - 연결된 edge 의 개수
  - 연결된 edge 의 weights 의 총합
  - …
- PageRank 는 그래프의 구조를 바탕으로 마디의 중요도를 정의합니다.

PageRank

- PageRank[1] 는 Google 의 검색 결과의 랭킹을 위하여 개발된 방법입니다.
  - 질의어를 포함한 웹문서는 수십만개입니다.
  - 그 중 가장 적절한 웹문서를 선택하기 위하여 웹페이지의 중요도를 정의해야 했습니다.

[1] Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The PageRank citation ranking: Bringing order to the web. Stanford InfoLab.

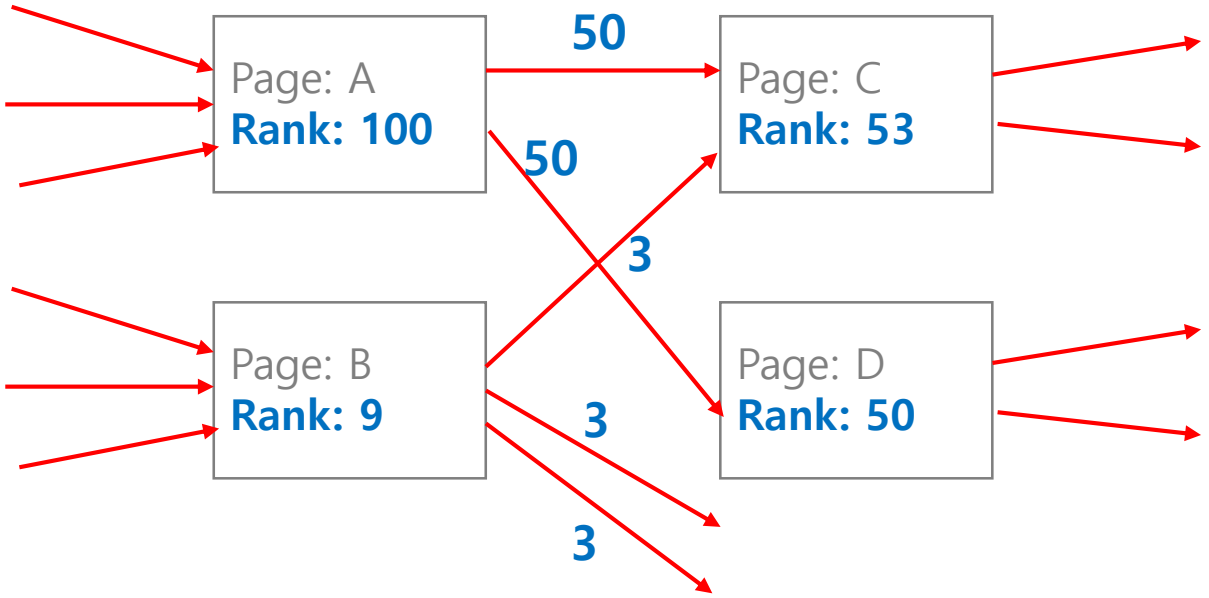
PageRank

- PageRank 는 웹공간의 hyper link 구조를 이용하여 중요도를 계산합니다.
  - 많은 back-links (자신으로 유입되는 링크)를 가진 페이지는 중요할 가능성이 높습니다.
  - 중요한 페이지의 hyper link 는 그렇지 않은 페이지보다 더 영향력이 있어야 합니다.

[1] Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The PageRank citation ranking: Bringing order to the web. Stanford InfoLab.

PageRank

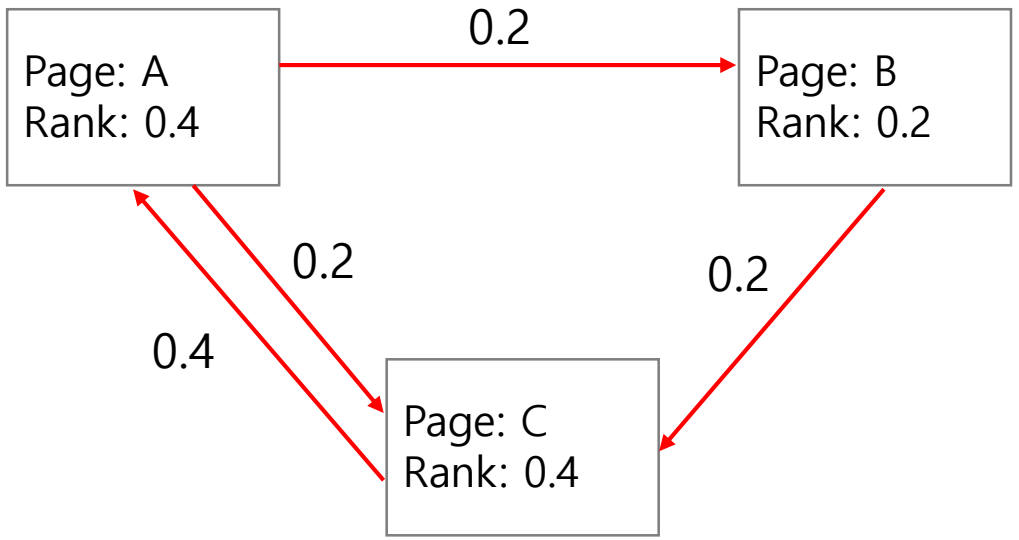
- PageRank 는 각 페이지의 중요도를, 연결된 페이지에 골고루 나눠줍니다.
  - 연결된 웹페이지에 투표를 하는 것과 같습니다.
  - 100 의 중요도를 지닌 페이지의 중요도는 두 페이지에 50 씩 나눠집니다.



[1] Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The PageRank citation ranking: Bringing order to the web. Stanford InfoLab.

PageRank

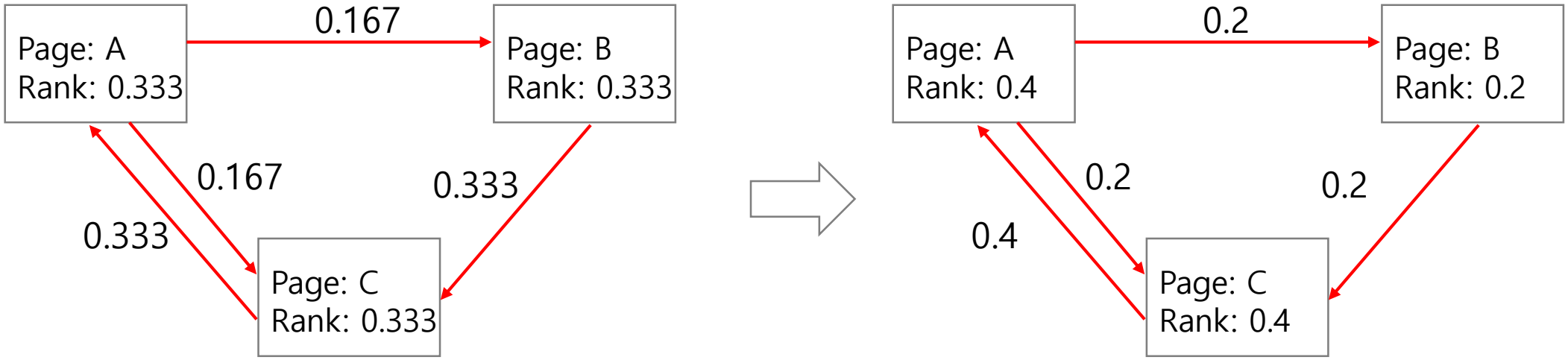
- PageRank 는 iterative 한 방법으로 각 페이지의 중요도를 계산합니다.
  - Rank 의 균형이 맞는 상태가 존재합니다.
  - 그러나 이 균형을 처음부터 알지는 못합니다.



[1] Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The PageRank citation ranking: Bringing order to the web. Stanford InfoLab.

PageRank

- PageRank 는 iterative 한 방법으로 각 페이지의 중요도를 계산합니다.
  - 초기화 때 모든 페이지에 같은 rank 를 부여합니다.
  - $PR(u) = c\sum_{v \in B_u} \frac{PR(v)}{N_v}$ 를 이용하여 모든 웹페이지 u 의 중요도를 계산합니다.



[1] Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The PageRank citation ranking: Bringing order to the web. Stanford InfoLab.

PageRank

- PageRank 는 iterative 한 방법으로 각 페이지의 중요도를 계산합니다.
  - Ant (random surfer) models 로 직관적인 설명이 가능합니다.
  - 개미들이 그래프 위의 마디를 무한히 이동하면 많은 개미가 몰리는 마디가 생겨납니다.
  - 몰려있는 개미의 상대적인 양이 마디들의 중요도입니다.
  - 웹공간에서는 각 페이지에 사용자들이 머무는 기대시간이기도 합니다.

PageRank

- Out links 가 없는 마디는 PageRank 가 제대로 작동하지 않게 만듭니다.
  - Out links 가 없으면 Iteration 이 지속될수록 누적된 개미의 양이 커집니다.
  - 웹페이지에서는 링크를 타고 유입된 사람이 다른 페이지를 더 이상 보지 않는 것과 같습니다.

PageRank

- Random jump 를 이용하여 문제를 해결합니다.
  - $PR(u) = c\sum_{v \in B_u} \frac{PR(v)}{N_v} + (1-c)\frac{1}{N}$
  - 각 페이지에서 c 만큼의 rank 만 out links 를 이용하고, $1-c$ 만큼은 모든 페이지에 임의로 이동합니다.
  - $(1-c)\frac{1}{N}$ 만큼 모든 페이지로부터 유입되었다는 의미입니다.
  - c 는 survival rate 입니다. $0 < c < 1$ 의 값을 이용합니다. (default = 0.85)

PageRank

- Algorithm

Input: [Graph $\boldsymbol{G}$, weight c]

Output: [PageRank $PR(u)$]

---

1. Initialize $PR(u)$ with $\frac{1}{N}$
2. While not converged, iterate

$$PR(u) = c \sum_{v \in B_u} \frac{PR(v)}{N_v} + (1 - c\,) \frac{1}{N}$$

PageRank

- $PR(u)$ 의 수렴은 매우 빠릅니다.
  - Iteration 마다 $PR(u)$ 의 차이는 지수승으로 줄어듭니다.
  - 322 M links 인 웹그래프에서도 iteration = 50 이면 충분합니다.

Personalized PageRank

- PageRank 는 웹페이지의 global importance 를 계산합니다.
  - 개인의 검색 이력 등의 정보를 이용하지 않은 중요도입니다.
- Bias 는 각 페이지에 대한 개인의 선호 (preference) 입니다.
  - Bias, $PR(u) = c\sum_{v \in B_u} \frac{PR(v)}{N_v} + (1-c)\frac{1}{N}$ 를 조절하면 개인의 성향 / 상황을 고려한 ranking 이 가능합니다.
  - $(1-c)\frac{1}{N}$ 은 선호가 없음을 의미합니다.

PageRank

- PageRank 에서 기억해야 할 점은 마디의 중요도를 정의하는 방식입니다.

중요한 마디로부터 많은 투표 (back-links)를 받는 마디가 중요한 마디

PageRank

- PageRank 의 구현체는 다양합니다.
- soygraph 는 텍스트 데이터의 handling 과 효율적인 계산을 위해 제안된 방법들을 구현하려는 프로젝트입니다.

https://github.com/lovit/soygraph

PageRank

```
from soygraph import dict_to_matrix

dd = {0: {1: 0.037, 55:0.025}, 1: {83:32, … }, … }
x = dict_to_matrix(dd)
print(type(x)) # <class 'scipy.sparse.csr.csr_matrix'>
print(x.shape) # (15097, 15097)

from soygraph.ranking import PageRank

pagerank = PageRank(
  damping_factor=0.85, max_iter=30, ranksum=1.0,
  verbose=True, converge_threshold=0.0001
)

rank_value = pagerank.rank(x)
rank_value = pagerank.rank(dd)
```


https://github.com/lovit/soygraph

HITS algorithm

- HITS[1] 는 웹검색의 ranking 을 위해 제안된 알고리즘입니다.
  - PageRank 와 비슷한 방식입니다.
  - 논문[1]에 기술된 알고리즘 전체의 디테일은 다릅니다만, ranking 방식은 비슷합니다.
  - HITS 는 각 마디(웹페이지)에 대하여 authority, hub 의 ranking 을 계산합니다.

[1] Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, *46*(5), 604-632.

HITS algorithm

- HITS 는 authority 와 hub 를 재귀적(recursive)으로 정의합니다.
  - $authority(u) = \sum_{v \in B_u} hub(v)$
  - $hub(u) = \sum_{v \in B_u} authority(v)$
- 이를 응용하면 $R(u) = \sum_{v \in B_u} R(v)$ 로 정의할 수 있습니다.

HITS algorithm

- 매 iteration 마다 normalization 이 필요합니다.
  - $R(u) = \sum_{v \in B_u} R(v)$ 형식은 그래프 전체의 rank sum 이 계속 증가합니다.
  - 매 iteration 마다 전체의 rank sum 이 유지되도록 정규화를 수행합니다.

HITS algorithm

```
from soygraph.ranking import HITS

hits = HITS(
    damping_factor=0.85, max_iter=30, ranksum=1.0,
    verbose=True, converge_threshold=0.0001
)

rank_value = hits.rank(x)
rank_value = hits.rank(dd)
```


https://github.com/lovit/soygraph

- PageRank 와 HITS 는 그래프의 마디에 대한 중요도를 정의 / 학습하는 방법입니다.
  - 두 알고리즘 모두 검색 결과의 ranking 을 위하여 제안되었지만, 다양한 분야에서 응용이 되고 있습니다.
  - 학습에 이용할 데이터가 목적에 적합한 그래프의 형태로 표현되기만 하면 알고리즘을 적용할 수 있습니다.

TextRank

- TextRank[1] 은 PageRank 를 이용하여 키워드 / 핵심문장을 추출합니다.
  - 텍스트로부터 단어 / 문장 그래프를 만든 뒤, PageRank를 적용합니다.

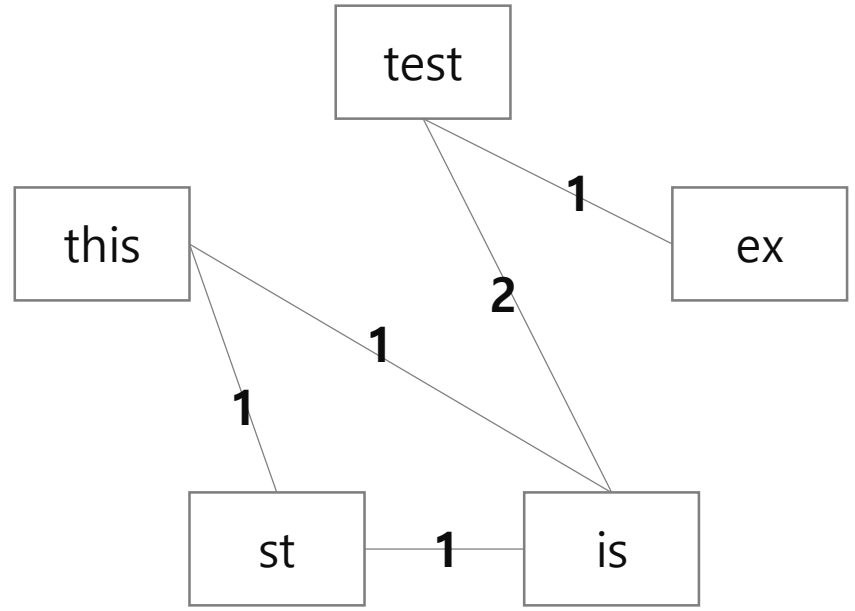
- Co-occurrence 가 있는 두 단어는 edge 로 연결됩니다.
  - Co-occurrence 는 window 안에 포함된 경우입니다.
  - Edge 의 값은 co-occurrence 입니다.

[1] Mihalcea, R., & Tarau, P. (2004). Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*.

TextRank

window = 1

doc = [[this, is, test, ex],
[this, st, is, test]]



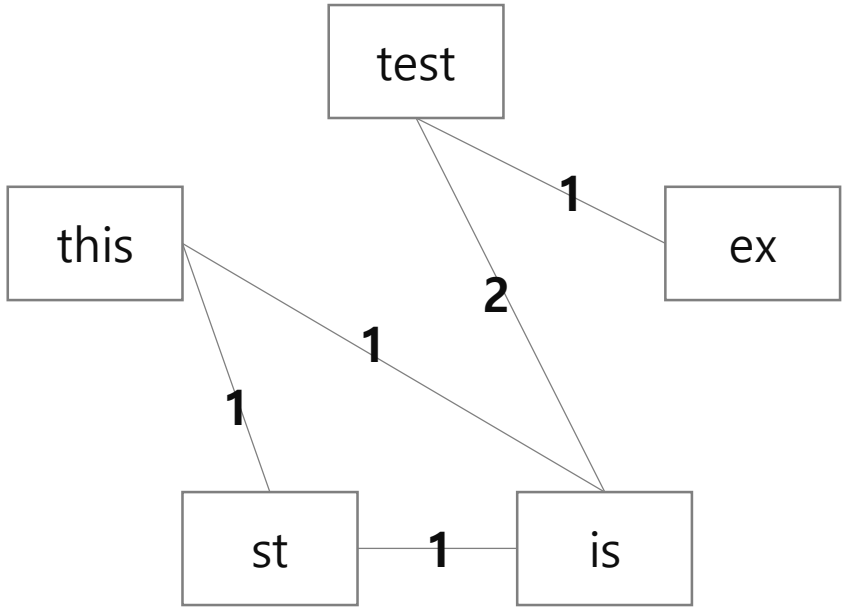
texts

word graph

TextRank

- $R(V_i) = (1 - d) + d \times \sum_{v_j \in In(V_i)} \frac{w_{ji}}{\sum_{v_k \in Out(V_j)} w_{jk}} R(V_j)$
  - Weighted PageRank version
  - $R(this) = 0.1 + 0.9 \times \left(\frac{1}{1} \times R(st) + \frac{1}{3} \times R(is)\right)$

    $with\ d = 0.9$

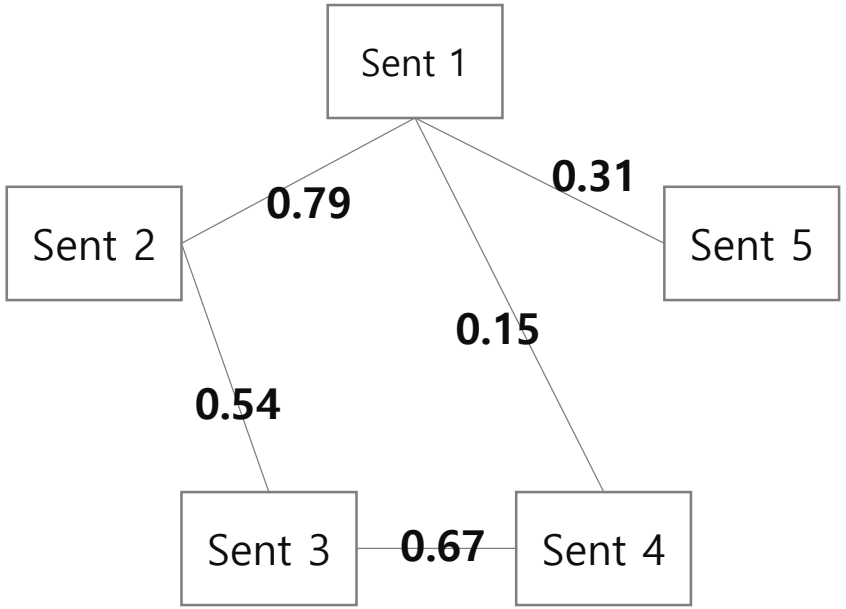


TextRank

- "중요한 단어 주위에 있는 단어는 중요하다" 라는 가정을 바탕으로 word graph 를 구성합니다.
- 빈도수가 많은 단어가 높은 rank 를 가질 가능성이 높습니다.

TextRank

- Sentence graph 는 모든 문장 간의 유사도를 edge weight 로 구성합니다.
  - 문장 간의 유사도는 임의의 유사도를 이용할 수 있습니다.



TextRank

- Sentence graph 는 모든 문장 간의 유사도를 edge weight 로 구성합니다.
  - 각 문장 $S_i = w_1^i, w_2^i, \dots, w_{N_i}^i$ 를 N_i 개의 단어 집합으로 생각합니다.
  - 문장 간 유사도는 아래의 식으로 정의합니다.

$$similarity(S_i, S_j) = \frac{\left|\{w_k | \, w_k \in S_i \& \, w_k \in S_j\}\right|}{\log(|S_i|) + \log(|S_j|)}$$

TextRank

- 많은 문장들과 높은 유사도로 연결이 되어 있는 문장은, 빈번히 등장하는 단어들을 다수 포함할 가능성이 높습니다.
- Sentence graph 를 이용하는 TextRank 는 빈번한 단어를 다수 포함한 문장을 핵심 문장으로 선택합니다.

TextRank

- 문장으로부터 word / sentence graph 를 만들어 PageRank 를 적용합니다.

```
from textrank import summarize_as_keywords

keywords = summarize_as_keywords(sents,  topk=50,
    tokenizer=lambda s:s.split(), min_count=10,
    min_cooccurrence=3,  verbose=True,  debug=True
)
```

```
[('재배포', 0.538140850221495),
 ('무단', 0.46747526750507146),
 ('금지', 0.3797005381404317),
 ('뉴시스', 0.1889406802065108),
 ('공감', 0.10557622894724966),
 ('저작권자', 0.07848823486967427),
  ... ]
```


https://github.com/lovit/textrank

TextRank

- 문장으로부터 word / sentence graph 를 만들어 PageRank 를 적용합니다.

```
from textrank import summarize_as_keysentences


keywords = summarize_as_keysentences (sents)
```


https://github.com/lovit/textrank

TextRank (gensim)

- Gensim 은 TextRank 를 이용한 summarizer 를 제공합니다.
  - Gensim 은 여러 문장에서 핵심 문장을 찾는 함수와
  - 여러 문서에서 핵심 문서를 찾는 함수를 제공합니다.

```
from gensim.summarization.summarizer import summarize
from gensim.summarization.summarizer import summarize_corpus

text = """Rice Pudding - Poem by Alan Alexander Milne
... What is the matter with Mary Jane?
... She's crying with all her might and main,
... And she won't eat her dinner - rice pudding again -
... What is the matter with Mary Jane? …"""

print(summarize(text))
```

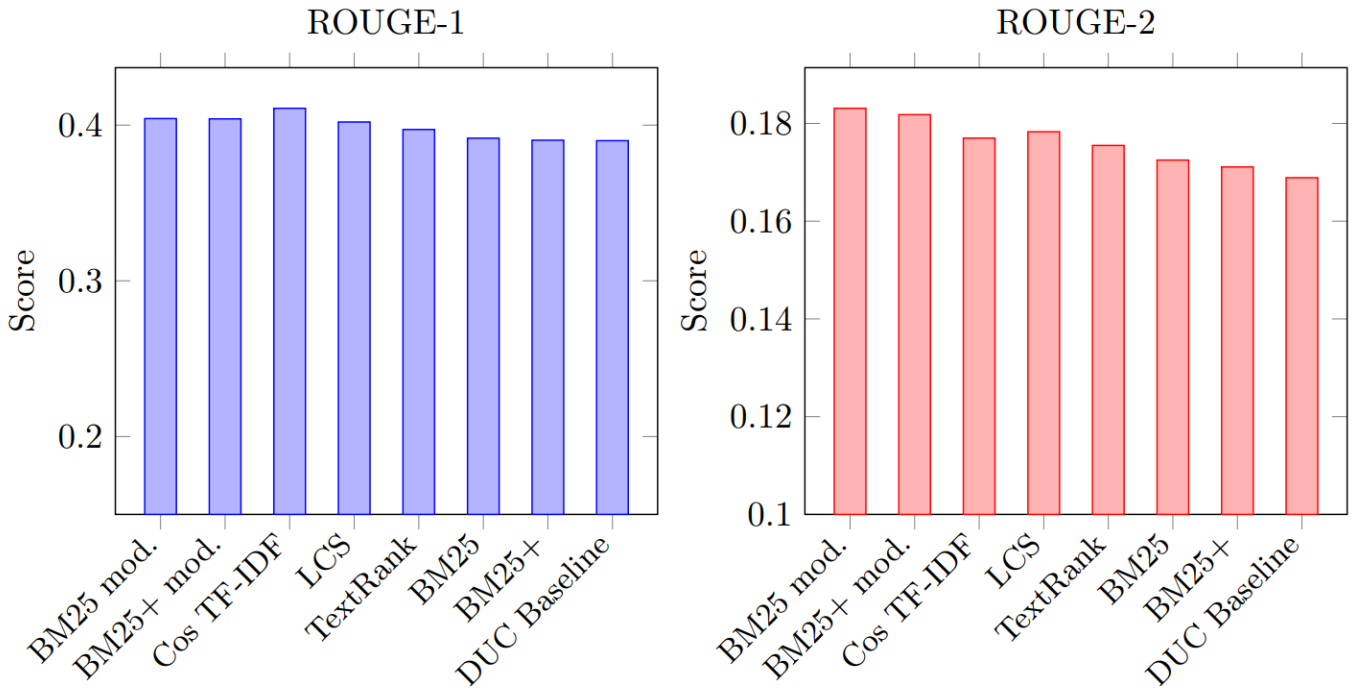
TextRank (gensim)

- Gensim 은 TextRank 를 이용한 summarizer 를 제공합니다.
  - Gensim 에 구현된 summarizer 는 TextRank 의 변형입니다.
  - Barrios et al., (2016) 은 TextRank 의 graph 를 만들 때, BM25+ 점수를 edge weight 로 이용하였습니다.

[1] Barrios, F., López, F., Argerich, L., & Wachenchauzer, R. (2016). Variations of the similarity function of textrank for automated summarization. *arXiv preprint arXiv:1602.03606*.

TextRank (gensim)

- TextRank 의 원형과 gensim summarizer 의 성능은 2002 Document Understanding Conference (DUC) 를 이용하여 ROGUE score 로 평가하였습니다.



[1] Barrios, F., López, F., Argerich, L., & Wachenchauzer, R. (2016). Variations of the similarity function of textrank for automated summarization. *arXiv preprint arXiv:1602.03606*.

LexRank

- Sentence graph 구성 시, 디테일이 TextRank 와 다릅니다.
  - 문장 간 유사도를 TF-IDF cosine 을 이용합니다.
- 문장 간 유사도의 최소값의 threshold 를 이용합니다.
  - 적절한 threshold 를 설정하면 sparse graph 를 만들 수 있으며,
  - 핵심 문장이 잘 추출됩니다.

[1] Erkan, G., & Radev, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, *22*, 457-479.

WordRank

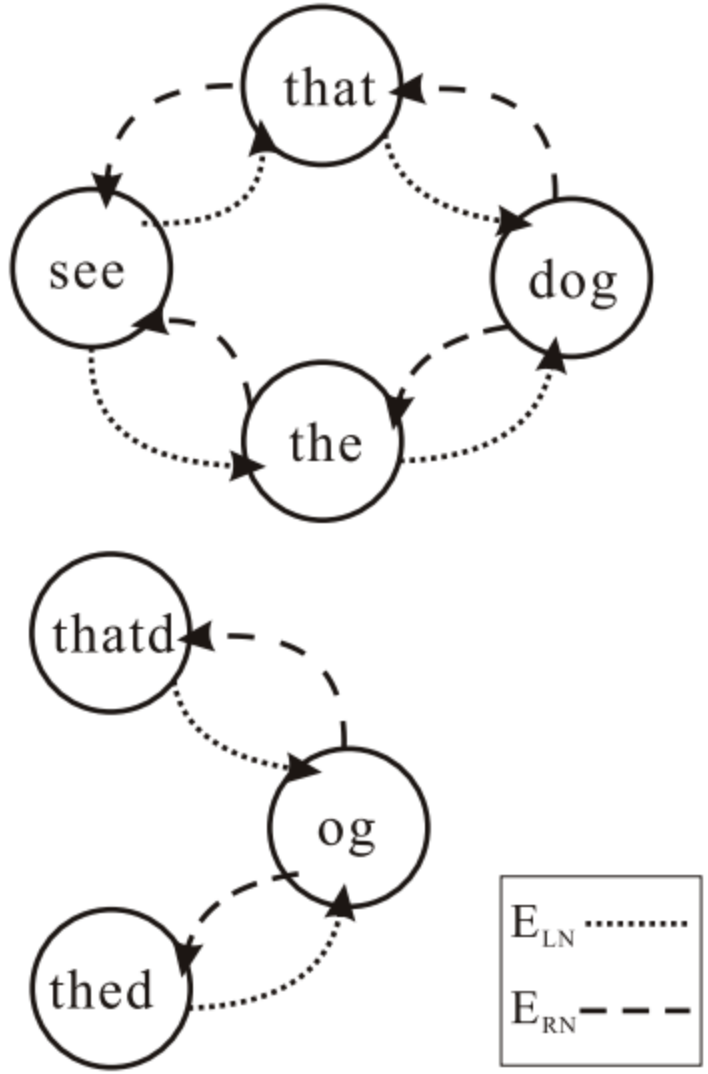
- TextRank 는 단어열로 분해된 문장으로부터 키워드를 추출합니다.
- Substring graph 를 이용하면 데이터 기반으로 단어 추출도 가능합니다.
  - WordRank 는 중국/일본어에서 단어를 추출하기 위하여 제안된 방법입니다.
  - "단어의 이웃은 단어이며, 단어가 아닌 substring 의 이웃은 잘못된 substring 이다"는 가정을 이용합니다.

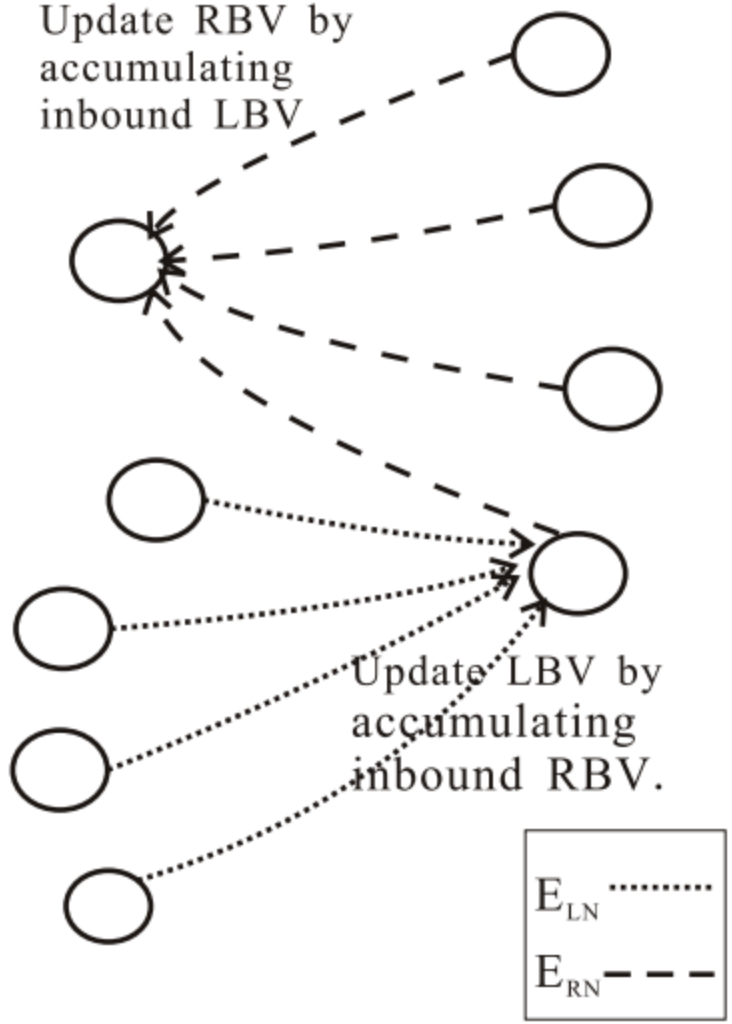
[1] Chen, S., Xu, Y., & Chang, H. (2011, August). A Simple and Effective Unsupervised Word Segmentation Approach. In AAAI.

do you see that dog
i dont see the dog
there is the dog

doyouseethatdog
idontseethedog
thereisthedog

e:	8	the:	3
o:	6	edo:	2
do:	5	hedo:	2
og:	3	seeth:	2
he:	3	hedog:	2
dog:	3	. . .	





(a) Segmented, unsegmented corpus and valid word hypotheses

(b) Illustration of the link structure (partial)

(c) Illustration of iterative updating

[1] Chen, S., Xu, Y., & Chang, H. (2011, August). A Simple and Effective Unsupervised Word Segmentation Approach. In AAAI.

KR-WordRank

- 한국어 텍스트에는 WordRank 적용이 어렵습니다.
  - 한국어에서 사용되는 글자 수가 적기 때문에 1음절 단어가 주요 단어로 추출
  - 띄어쓰기 정보를 무시하면 잘못된 단어후보가 단어로 추출될 수 있습니다.
    - "너는지난주" vs "너는 지난주"
  - 추출해야 하는 단어는 어절의 왼쪽에 위치하는 substring 뿐입니다.
    - "트와이스는"

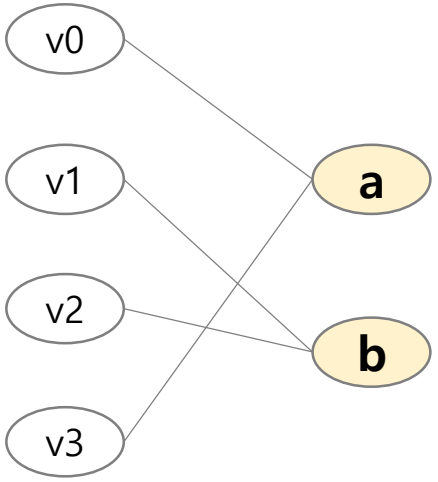
KR-WordRank

- Packages
  - https://github.com/lovit/kr-wordrank
  - > pip install krwordrank

```
from krwordrank.word import KRWordRank

min_count = 5   # 단어의 최소 출현 빈도수 (그래프 생성 시)
max_length = 10 # 단어의 최대 길이
wordrank_extractor = KRWordRank(min_count, max_length)

texts = ['예시 문장 입니다', '여러 문장의 list of str 입니다', ... ]
keywords, rank, graph = wordrank_extractor.extract(texts)
```

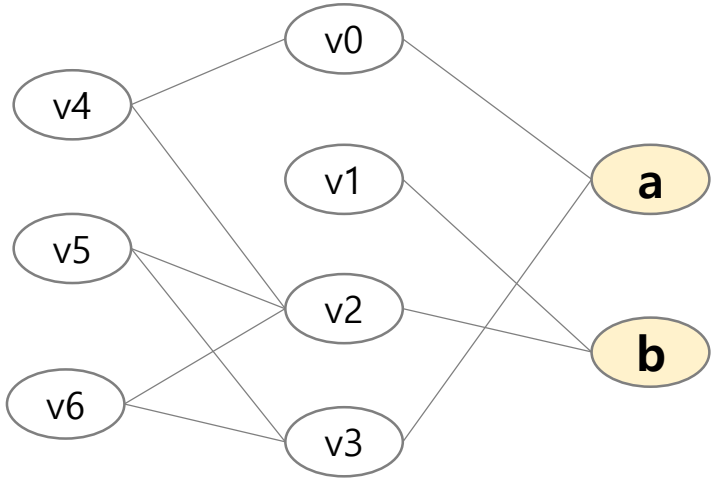
Which nodes are similar?

- 두 마디 (a, b)의 이웃이 유사하면 (a, b)는 비슷하다 정의할 수 있습니다.
  - (a, b) 의 이웃 벡터에 대한 Cosine / Jaccard similarity 는 0입니다.
  - (a, b) 를 문서, (v0, … v3) 을 단어로 생각하면 (a, b) 는 동일한 단어를 공유하지 않습니다.



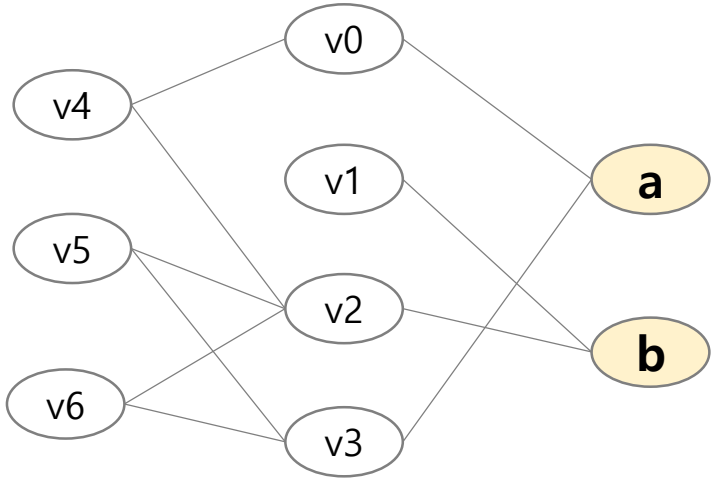
Which nodes are similar?

- 두 마디 (a, b)의 이웃이 유사하면 (a, b)는 비슷하다 정의할 수 있습니다.
  - (v2, v3) 는 공유하는 이웃이 많기 때문에 비슷한 마디입니다.
  - (v2, v3) 의 유사성을 고려하면 (a, b) 도 비슷한 마디여야 합니다.
  - 이웃의, 이웃의, … 이웃을 고려합니다.



SimRank

- SimRank[1] 는 그래프의 구조를 반영하여 마디 간 유사도를 정의합니다.



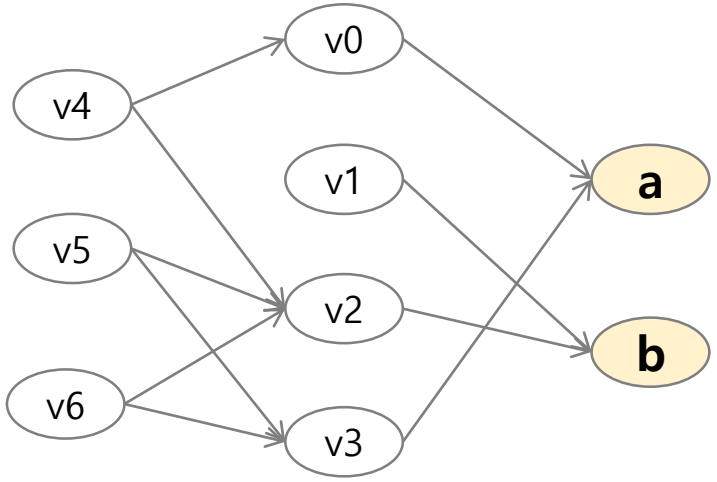
[1] Jeh, G., & Widom, J. (2002, July). SimRank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD*

SimRank

- SimRank 도 재귀적으로 마디 간 유사도를 정의합니다.

$$S(a,b) = \frac{C}{|I(a)||I(b)|} \sum_{i \in I(a)} \sum_{j \in I(b)} S(i,j)$$

$I(a)$: 마디 a 의 inbounds
$I(b)$: 마디 b 의 inbounds

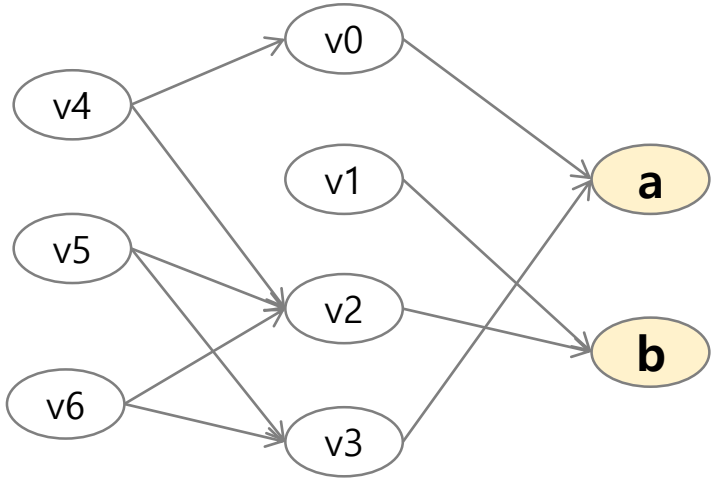


SimRank

- SimRank 의 학습 역시 iterative 하게 진행됩니다.

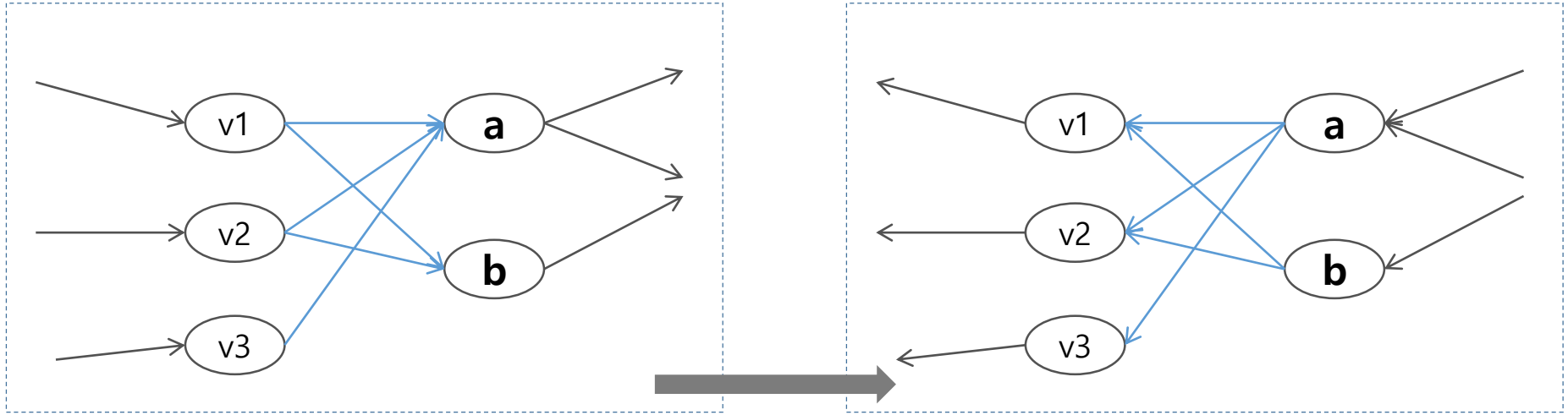
$$S_{k+1}(a,b) = \frac{C}{|I(a)||I(b)|} \sum_{i \in I(a)} \sum_{j \in I(b)} S_k(i,j)$$

$I(a)$: 마디 a 의 inbounds
$I(b)$: 마디 b 의 inbounds
$S_0(a,b) = 1$ if $a = b$ else 0



SimRank 의 의미

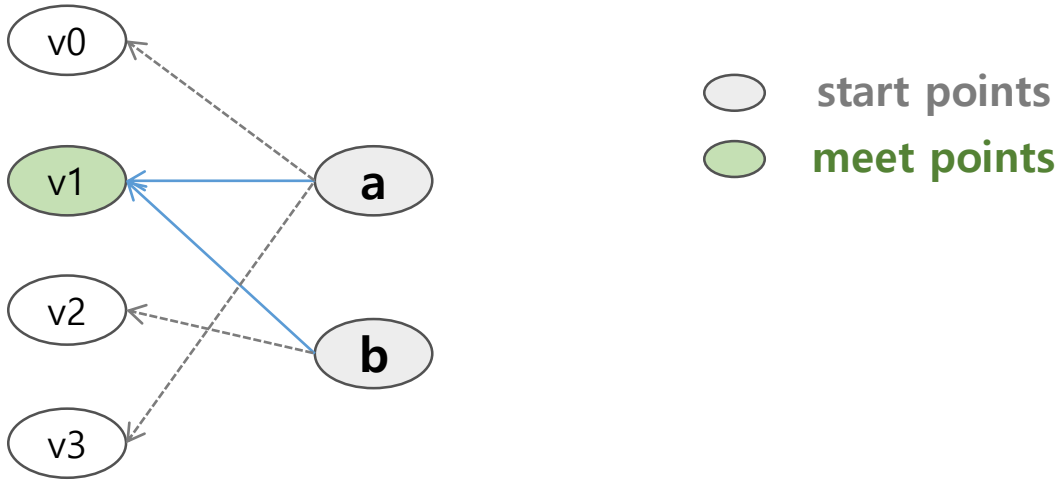
- SimRank 는 reversed graph 에서의 random walker 로 해석합니다.
  - k 번의 iteration 을 통하여 학습된 SimRank, $S_k(a, b)$ 는 reversed graph 에서 (a, b) 를 출발한 random walker 가 k step 안에 만날 확률입니다.



reversed random walker model

SimRank 의 의미

$$Sim_{k+1}(a,b) = \frac{C|_{c=1}}{|I(a)||I(b)|} \sum_{i \in I(a)} \sum_{j \in I(b)} Sim_k(i,j)$$



At 1 step iteration,

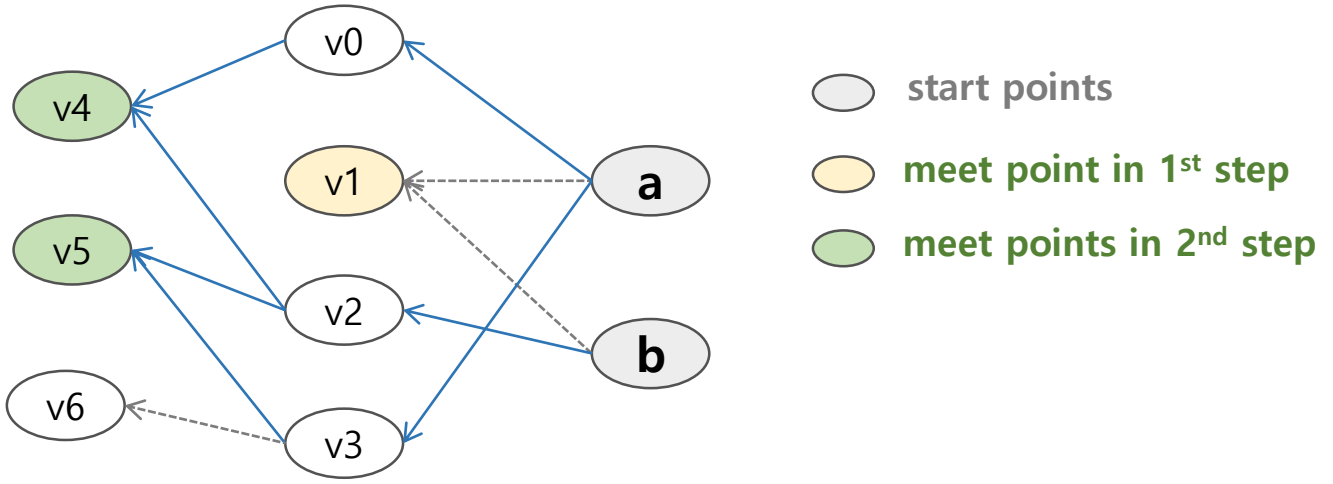
'a' has 3 paths, 'b' has 2 paths, (1 commons)

- Prob. of meeting at v1 $= \frac{1}{3} \times \frac{1}{2} = \frac{1}{6}$

- Prob. of meeting within 1 step$= \frac{1}{6}$

SimRank 의 의미

$$Sim_{k+1}(a,b) = \frac{C|_{c=1}}{|I(a)||I(b)|} \sum_{i \in I(a)} \sum_{j \in I(b)} Sim_k(i,j)$$



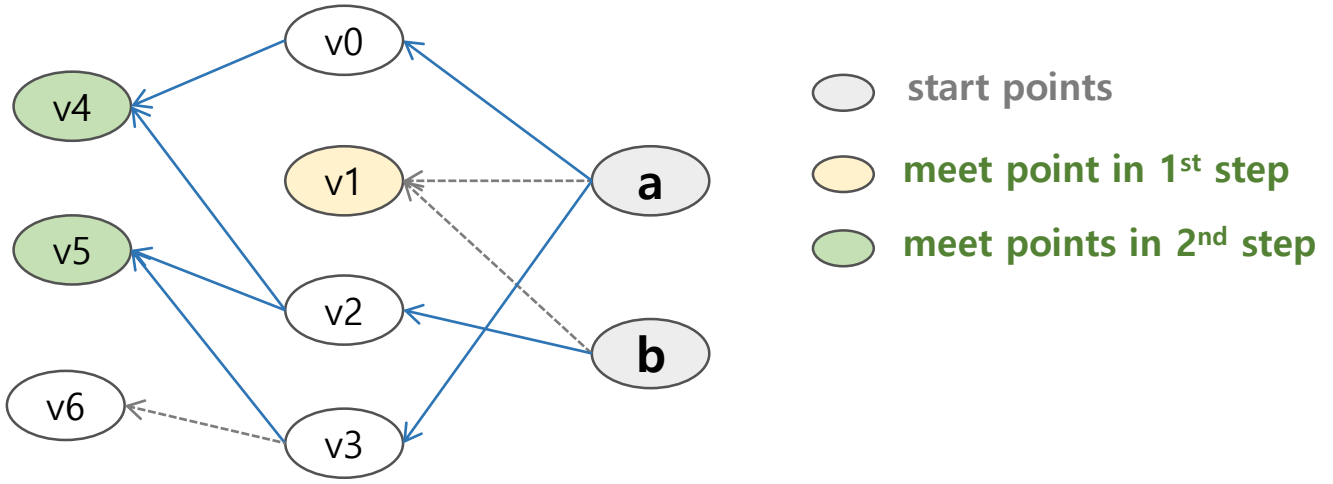
At 2 step iteration,

- Prob. of meeting at v4 | {a – v0 – v4 // b – v2 – v4 } $= \left(\frac{1}{3} \times 1\right) \times \left(\frac{1}{2} \times \frac{1}{2}\right) = \frac{1}{12}$
- Prob. of meeting at v5 | {a – v3 – v5 // b – v2 – v5 } $= \left(\frac{1}{3} \times \frac{1}{2}\right) \times \left(\frac{1}{2} \times \frac{1}{2}\right) = \frac{1}{24}$
- Prob. of meeting at 2 steps$= \frac{3}{24}$
- Prob. of meeting within 2 steps$= \frac{1}{6} + \frac{3}{24} = \frac{7}{24}$

SimRank 의 의미

- Survival rate C 를 고려합니다.

$$Sim_{k+1}(a,b) = \frac{C|_{c=1}}{|I(a)||I(b)|} \sum_{i \in I(a)} \sum_{j \in I(b)} Sim_k(i,j)$$

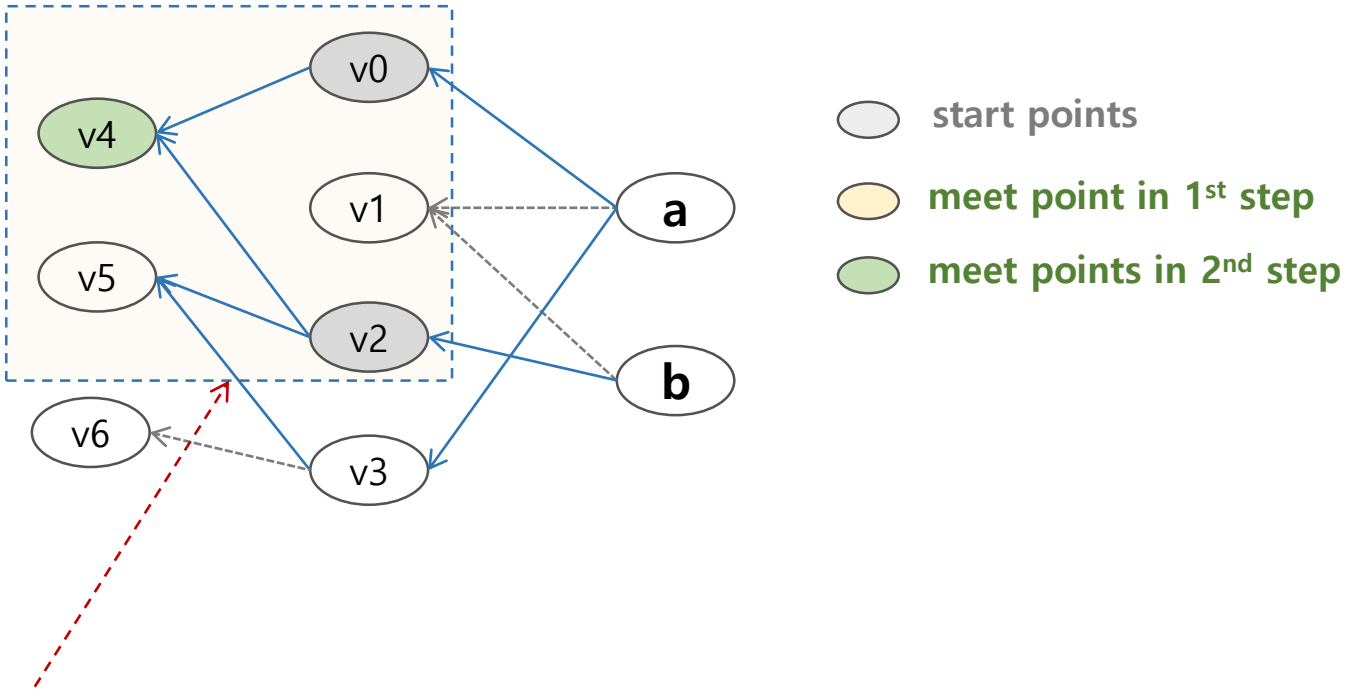


At 2 step iteration with decaying factor

- Prob. of meeting within 2 steps= $c\frac{1}{6} + c^2\frac{3}{24}$

SimRank 의 의미

$$Sim_1(a,b) = \frac{C|_{c=1}}{|I(a)||I(b)|} \sum_{i \in I(a)} \sum_{j \in I(b)} Sim_0(i,j)$$

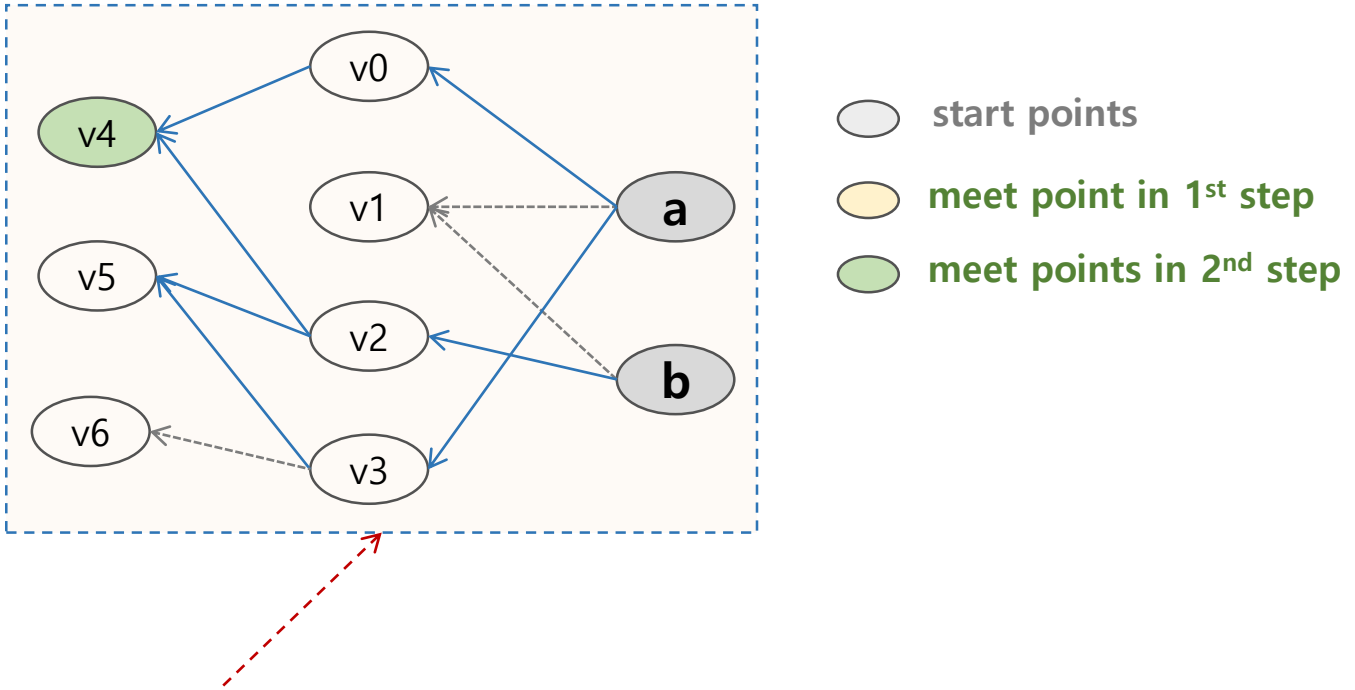


Iteration 1 에서 $Sim_1(v0, v2)$ 가 업데이트되지만,

$Sim_1(v0, v2)$ 가 $Sim_1(a, b)$ 까지 전달되지는 않습니다

SimRank 의 의미

$$Sim_1(a,b) = \frac{C|_{c=1}}{|I(a)||I(b)|} \sum_{i \in I(a)} \sum_{j \in I(b)} Sim_0(i,j)$$

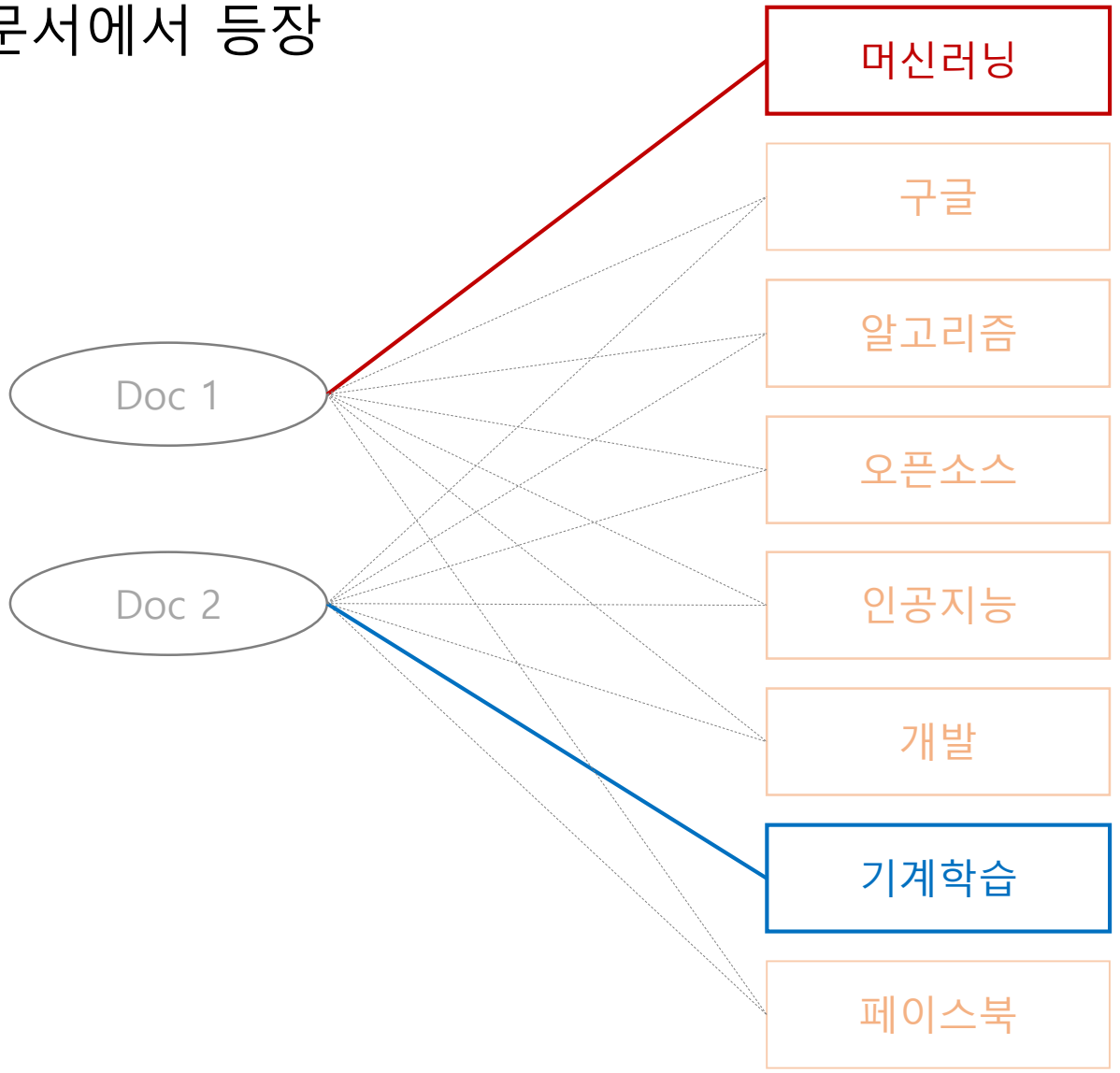


Iteration 2 에서 $Sim_1(v0, v2)$ 고려되어 $Sim_2(a,b)$ 에 업데이트 됩니다.

SimRank 의 의미

- Term – frequency (bipartite) graph 에서는 같은 문서에 등장하지 않은 단어더라도 유사한 토픽에서 등장하는 단어를 찾을 수 있습니다.

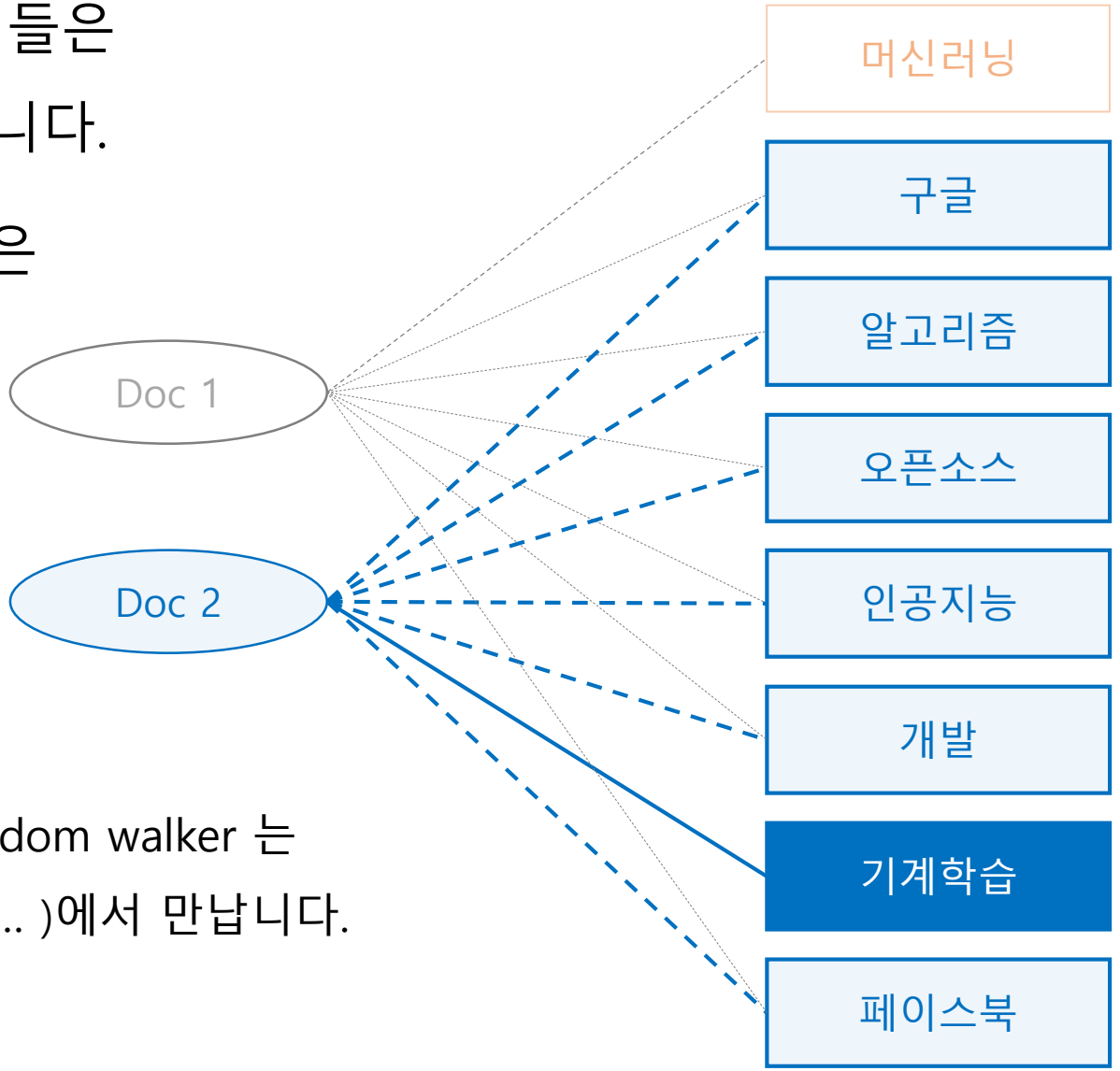
- (머신러닝, 기계학습) 은 같은 문서에서 등장한 적이 없습니다.



- “머신러닝”과 함께 등장한 단어들은 “기계학습”과도 함께 등장했습니다.
- “머신러닝”, “기계학습”의 이웃은 서로 비슷합니다.

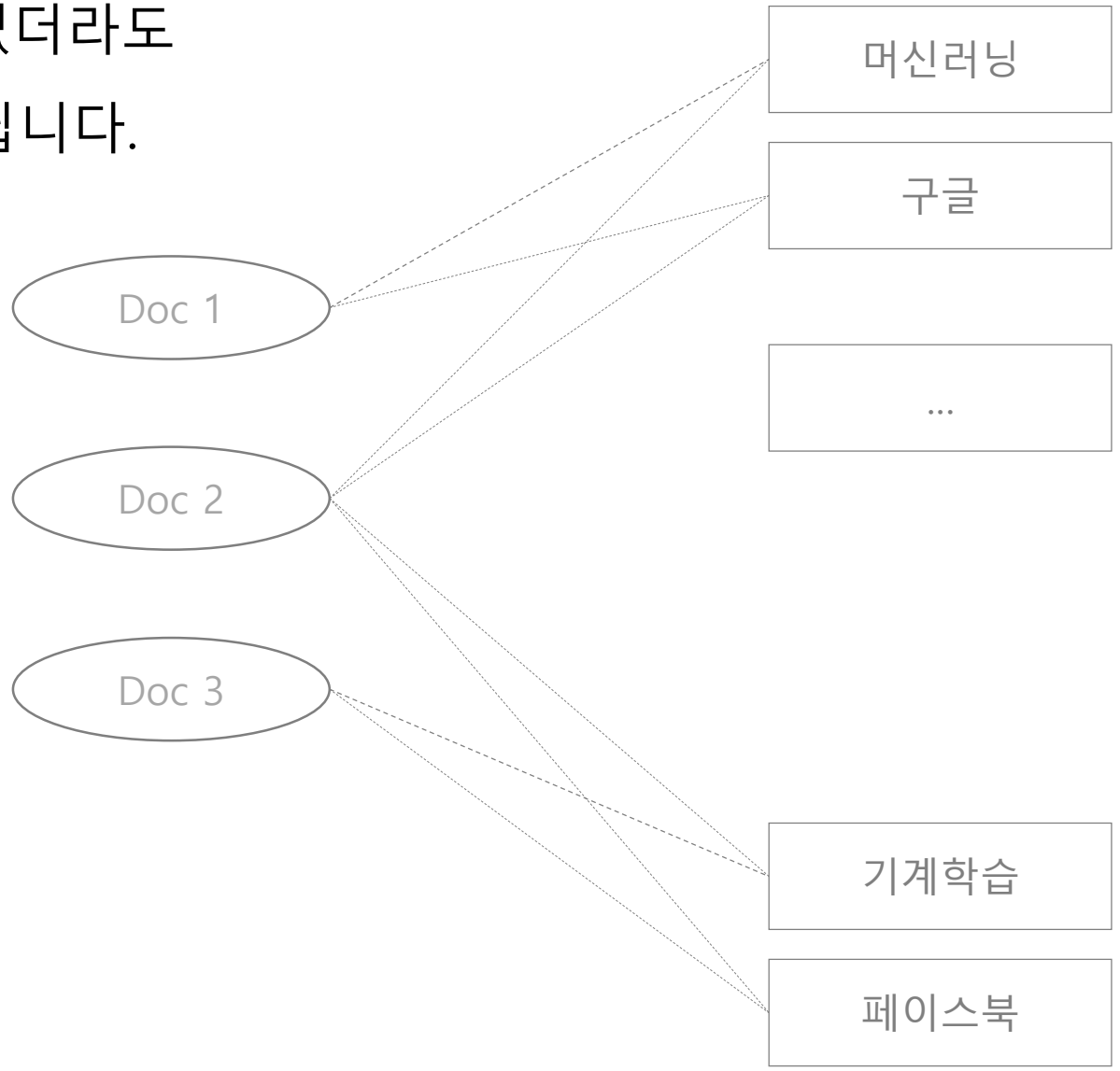
- (머신러닝, 기계학습) 에서 출발한 random walker 는 2 step 후 (구글, 알고리즘, 오픈소스, …)에서 만납니다.

- “머신러닝”과 함께 등장한 단어들은 “기계학습”과도 함께 등장했습니다.
- “머신러닝”, “기계학습”의 이웃은 서로 비슷합니다.



- (머신러닝, 기계학습) 에서 출발한 random walker 는 2 step 후 (구글, 알고리즘, 오픈소스, …)에서 만납니다.

- 문서 1과 3은 공통된 단어가 없더라도
  문서 2를 통하여 유사도를 지닙니다.



SimRank

- Doc2Vec 은 의 단어 유사성을 바탕으로, 문서간 유사도를 정의합니다.
  - 공통된 단어가 없더라도 비슷한 단어들이 포함된 두 개의 문서의 document vector는 비슷합니다.
- SimRank 은 Doc2Vec 과 유사합니다.
  - 유사한 단어가 포함된 두 개의 문서는 공통된 단어가 없더라도 유사합니다.

Small world effect

- Small world effect 는 Stanley Milgram 의 six degrees of separation 이론으로 설명됩니다.
  - 한 나라 안에서 모든 사람들은 여섯 단계를 걸치면 서로 아는 사이입니다.
  - 네트워크에서는 몇 단계만 거치면 similarity 가 존재합니다.

Small world effect

- SimRank 의 similarity matrix 는 몇 번의 iteration 만으로도 sparsity 가 급격히 내려갑니다.
  - 영화 1,949개, 출연 배우 7,7776명, 매우 sparse한 네트워크

# Iteration	# of visited actors	# of movie similarity > 0
1	7.158543	18.4274
2	18.4274	229.7912
3	112.9764	679.4854
4	**229.7912**	**1103.132**
5	1013.583	1513.811
6	679.4854	1635.561
7	2686.14	1662.276
8	1103.132	1668.141
9	4380.891	1669.592

SimRank

- 모든 node pairs 간의 similarity 가 필요한 경우는 적습니다.
  - Lee et al., (2012) 에서는 한 query node 의 top k 개의 유사 마디만을 계산하는 방법을 제안하였습니다.
  - soygraph 에서 SingleVectorSimRank 로 구현되어 있습니다.

Lee, P., Lakshmanan, L. V., & Yu, J. X. (2012, April). On top-k structural similarity search. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on* (pp. 774-785). IEEE.

SimRank

- SimRank 는 topically similar 한 단어들을 학습합니다.

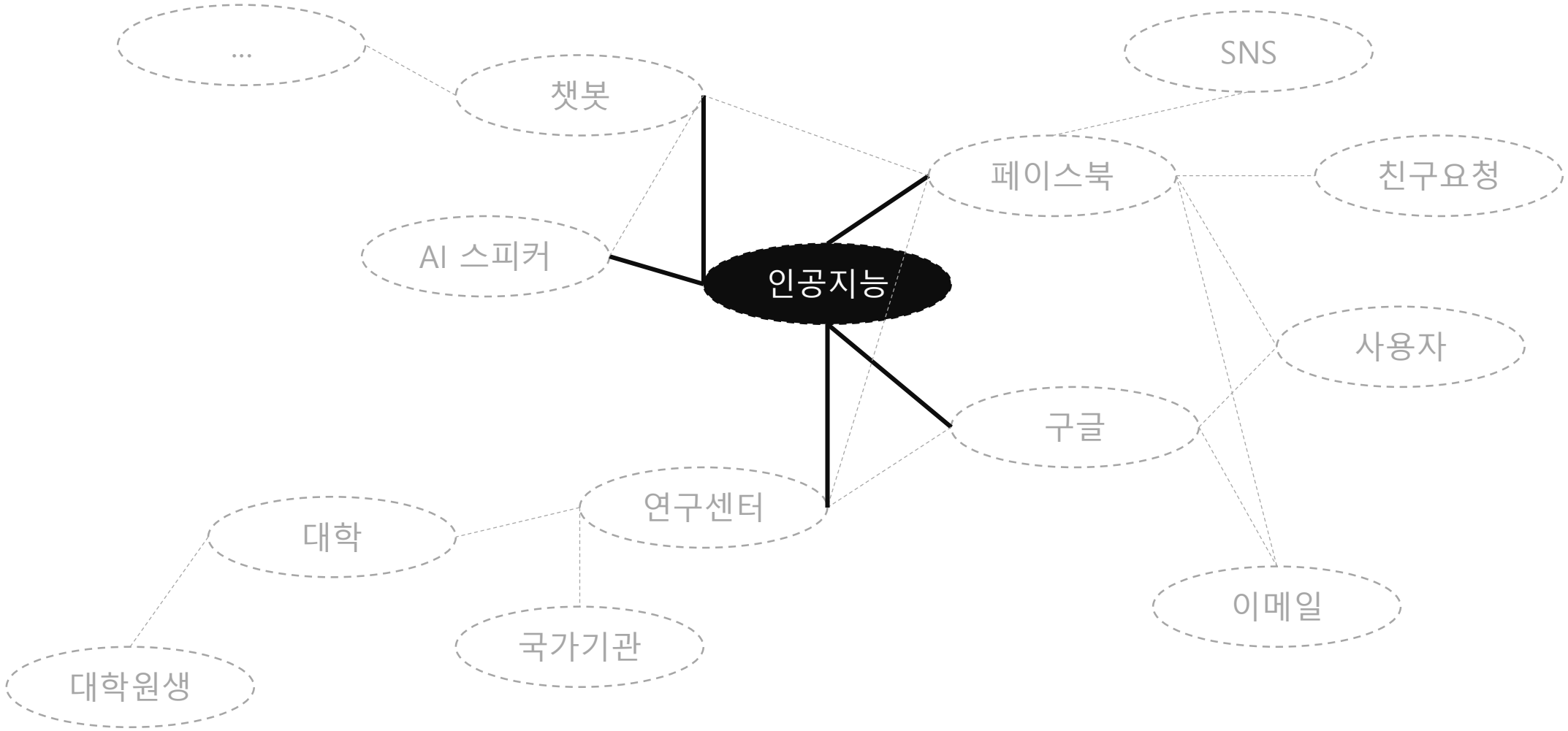
```
from soygraph.similarity import SingleVectorSimRank
from soygraph import DictGraph

g = DictGraph(dd)
simrank = SingleVectorSimRank(g)
similars = simrank.most_similar(vocab2node['아이오아이'], max_iter=4, topk=30)
```

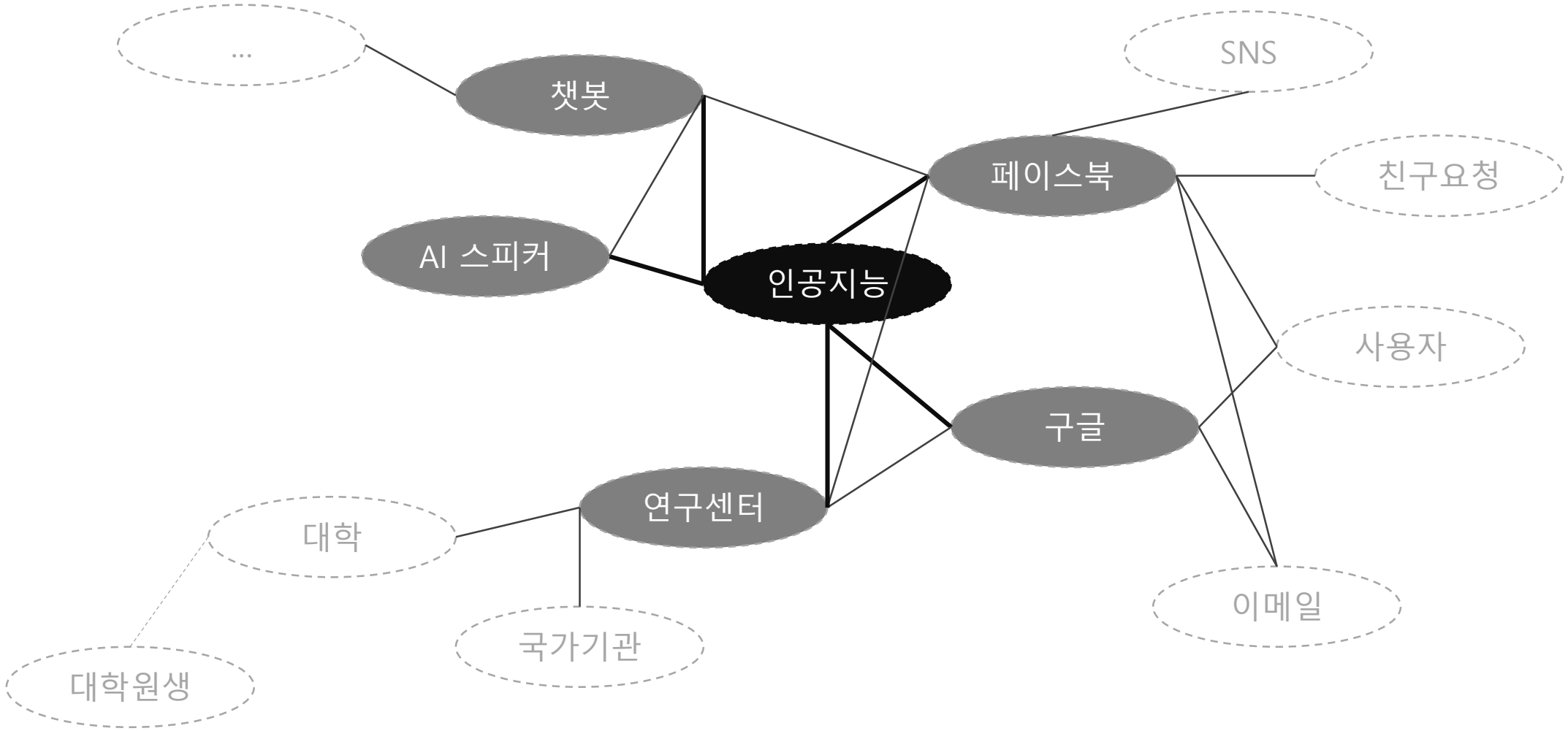
```
너무너무너무 박진영 빅브레인 완전체 신용재 오블리스
갓세븐 엠카운트다운 중독성 잠깐 세븐 다비치 상큼 소녀들
선의 산들 수록곡 프로듀스101 펜타곤 열창 타이틀곡 엠넷
본명 박소라 음원차트 깜찍 이진희 불독 키미 음악방송
```

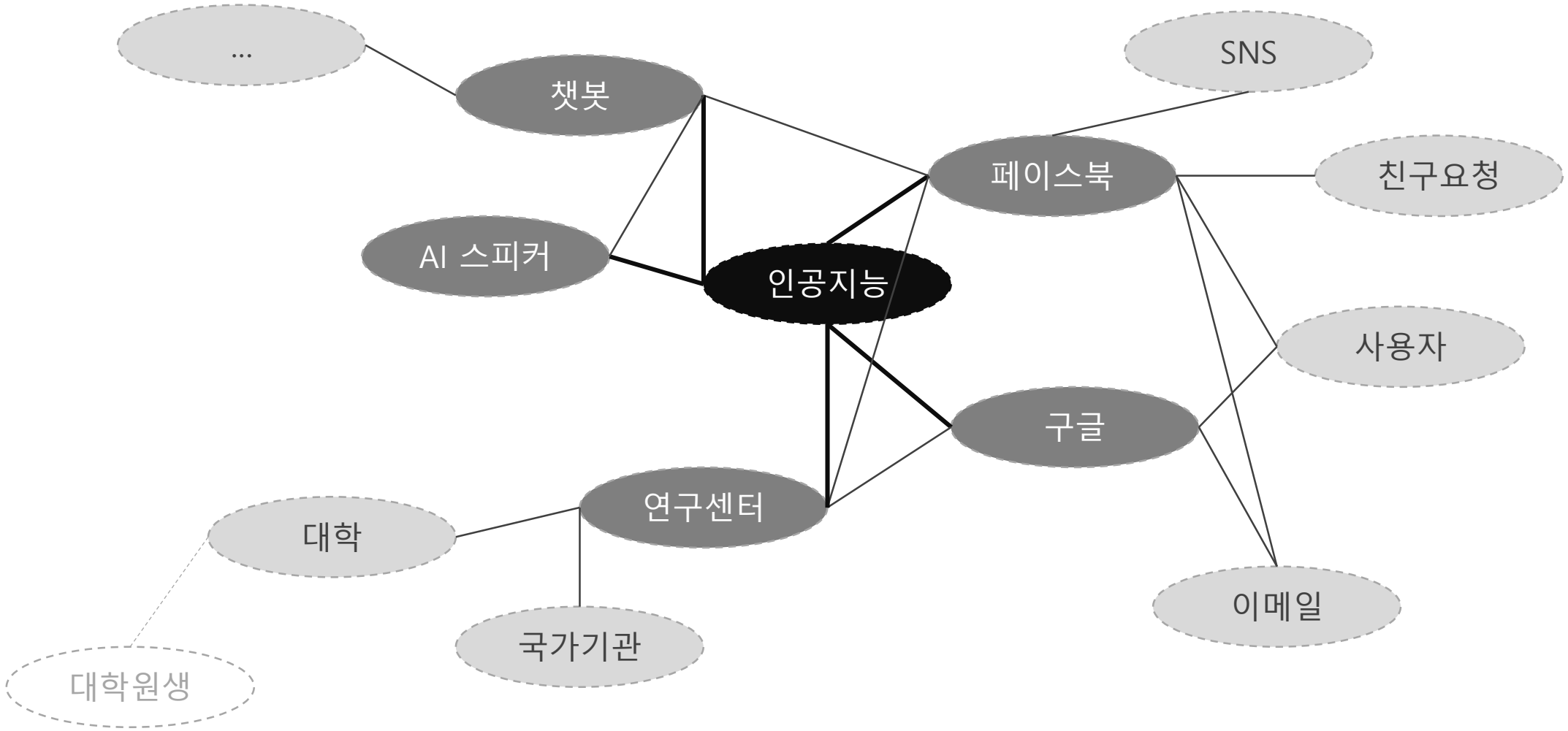
Random Walk with Restart (RWR)

- RWR[1,2] 은 근처에 있는 마디를 탐색합니다.
  - Random walker 의 모델을 이용하여 local nodes 를 탐색합니다.
  - $R_{k+1}(u\,|v) = c \cdot W \cdot R_k(u\,|v) + (1 - c)e_v$

[1] Tong, H., Faloutsos, C., & Pan, J. Y. (2006). Fast random walk with restart and its applications.
[2] Pan, J. Y., Yang, H. J., Faloutsos, C., & Duygulu, P. (2004, August). Automatic multimedia cross-modal correlation discovery. In *Proceedings of ACM SIGKDD*


...
챗봇
SNS
페이스북
친구요청
AI 스피커
인공지능
사용자
구글
연구센터
대학
이메일
국가기관
대학원생


...
챗봇
SNS
페이스북
친구요청
AI 스피커
인공지능
사용자
구글
연구센터
대학
이메일
국가기관
대학원생


...
챗봇
SNS
페이스북
친구요청
AI 스피커
인공지능
사용자
구글
연구센터
대학
이메일
국가기관
대학원생

Random Walk with Restart (RWR)

- RWR 과 SimRank 는 이웃한 마디들을 유사한 마디로 학습합니다.
- RWR 은 홀수 개의 edges 로 연결된 마디도 유사도가 정의되지만,
  SimRank 는 반드시 짝수 개의 edges 로 연결된 마디만 유사도를 지닙니다.

Small-world phenomenon

- SimRank, RWR 모두 기본 알고리즘은 계산 비용이 비쌉니다.
  - 그래프의 마디들은 몇 단계만 거쳐도 연결이 됩니다.
  - SimRank 의 $S_k(a, b) > 0$, RWR 의 $R_k(u \mid v) > 0$ 인 마디가 늘어납니다.
- 큰 그래프에 적용할 경우에는 다양한 계산 최적화 방법들을 이용합니다.

Random Walk with Restart (RWR)

- RWR 는 topically similar 한 단어들을 학습합니다.

```
from soygraph.similarity import RandomWalkWithRestart


rwr = RandomWalkWithRestart(x)
similars = rwr.most_similar(vocab2node['아이오아이'], max_iter=6, topk=30)
```

```
빅브레인 너무너무너무 오블리스 신용재 갓세븐 아이오아이
엠카운트다운 다비치 세븐 완전체 박진영 펜타곤 산들 중독성
엠넷 열창 잠깐 깜찍 타이틀곡 상큼 소녀들 몬스타엑스 일산동구
키미 불독 프로듀스 소라 방탄소년단 형은 파워풀
```