Covering problems

BOSTON UNIVERSITY

Prototype problems: Covering problems

- Setting:
  - Universe of N elements $U = \{U_1,\ldots,U_N\}$
  - A set of n sets $S = \{s_1,\ldots,s_n\}$
  - Find a collection C of sets in S (C subset of S) such that $\cup_{c \in C} c$ contains many elements from U
- Example:
  - U: set of documents in a collection
  - s_i: set of documents that contain term t_i
  - Find a collection of terms that cover most of the documents

evimaria@cs.bu.edu

Prototype covering problems

- **Set cover problem:** Find a small collection C of sets from S such that all elements in the universe U are covered by some set in C
- **Best collection problem:** find a collection C of k sets from S such that the collection covers as many elements from the universe U as possible
- Both problems are NP-hard
- Simple approximation algorithms with provable properties are available and very useful in practice

evimaria@cs.bu.edu

Set-cover problem

- Universe of N elements $U = \{U_1, \ldots, U_N\}$
- A set of n sets $S = \{s_1, \ldots, s_n\}$ such that $\cup_i s_i = U$

- **Question:** Find the smallest number of sets from S to form collection C (C subset of S) such that $\cup_{c \in C} c = U$

- The set-cover problem is **NP-hard** (what does this mean?)

evimaria@cs.bu.edu

Trivial algorithm

evimaria@cs.bu.edu

Trivial algorithm

- Try all subcollections of **S**

evimaria@cs.bu.edu

Trivial algorithm

- Try all subcollections of **S**

evimaria@cs.bu.edu

Trivial algorithm

- Try all subcollections of **S**
- Select the smallest one that covers all the elements in **U**

evimaria@cs.bu.edu

Trivial algorithm

- Try all subcollections of **S**
- Select the smallest one that covers all the elements in **U**

BOSTON UNIVERSITY
evimaria@cs.bu.edu

Trivial algorithm

- Try all subcollections of **S**
- Select the smallest one that covers all the elements in **U**
- The running time of the trivial algorithm is $O(2^{|S|}|U|)$

BOSTON UNIVERSITY

evimaria@cs.bu.edu

Trivial algorithm

- Try all subcollections of **S**
- Select the smallest one that covers all the elements in **U**
- The running time of the trivial algorithm is $O(2^{|S|}|U|)$

evimaria@cs.bu.edu

Trivial algorithm

- Try all subcollections of **S**
- Select the smallest one that covers all the elements in **U**
- The running time of the trivial algorithm is $O(2^{|S|}|U|)$
- This is way too slow

evimaria@cs.bu.edu

Greedy algorithm for set cover

- Select first the largest-cardinality set **s** from **S**
- Remove the elements from **s** from **U**
- Recompute the sizes of the remaining sets in **S**
- Go back to the first step

evimaria@cs.bu.edu

As an algorithm

- X = U
- C = {}
- **while** X is not empty **do**
  - For all $s \epsilon S$ let a_s=|s intersection X|
  - Let s be such that a_s is maximal
  - C = C U {s}
  - X = X\ s

BOSTON UNIVERSITY

evimaria@cs.bu.edu

How can this go wrong?

- No global consideration of how good or bad a selected set is going to be

evimaria@cs.bu.edu

How good is the greedy algorithm?

evimaria@cs.bu.edu

How good is the greedy algorithm?

- Consider a minimization problem
  - In our case we want to minimize the **cardinality** of set **C**
- Consider an instance **I**, and cost **$a^*(I)$** of the optimal solution
  - **$a^*(I)$**: is the minimum number of sets in **C** that cover all elements in **U**
- Let **a(I)** be the cost of the approximate solution
  - **a(I)**: is the number of sets in **C** that are picked by the greedy algorithm
- An algorithm for a minimization problem has approximation factor **F** if for all instances **I** we have that

$$a(I) \leq F \times a^*(I)$$

- **Can we prove any approximation bounds for the greedy algorithm for set cover ?**

evimaria@cs.bu.edu

How good is the greedy algorithm for set cover?

How good is the greedy algorithm for set cover?

- ***(Trivial?) Observation***: The greedy algorithm for set cover has approximation factor $F = s_{max}$, where s_{max} is the set in S with the largest cardinality

evimaria@cs.bu.edu

How good is the greedy algorithm for set cover?

- ***(Trivial?) Observation***: The greedy algorithm for set cover has approximation factor $F = s_{max}$, where s_{max} is the set in S with the largest cardinality
- **Proof:**
  - $a^*(I) \geq N/|s_{max}|$ or $N \leq |s_{max}| a^*(I)$
  - $a(I) \leq N \leq |s_{max}| a^*(I)$

BOSTON UNIVERSITY

evimaria@cs.bu.edu

How good is the greedy algorithm for set cover? A tighter bound

- The greedy algorithm for set cover has approximation factor $F = O(\log |s_{max}|)$

- **Proof**: (From CLR "Introduction to Algorithms")

evimaria@cs.bu.edu

Best-collection problem

- Universe of **N** elements $U = \{U_1,\ldots,U_N\}$
- A set of **n** sets $S = \{s_1,\ldots,s_n\}$ such that $\cup_i s_i = U$
- **Question:** Find the a collection **C** consisting of **k** sets from **S** such that $f(C) = |\cup_{c \in C} c|$ is **maximized**
- The best-colection problem is NP-hard
- Simple approximation algorithm has approximation factor $F = (e-1)/e$

evimaria@cs.bu.edu

Greedy approximation algorithm for the best-collection problem

- $C = \{\}$
- **for every** set s in S and **not** in C compute the gain of s:

$$g(s) = f(C \cup \{s\}) - f(C)$$

- Select the set s with the **maximum** gain
- $C = C \cup \{s\}$
- **Repeat until** C has k elements

Basic theorem

- The **greedy** algorithm for the best-collection problem has approximation factor **$F = (e-1)/e$**

- **C^* : optimal** collection of cardinality **k**
- **C :** collection output by the **greedy** algorithm
- **$f(C) \geq (e-1)/e \times f(C^*)$**

BOSTON UNIVERSITY

evimaria@cs.bu.edu

Reference

Reference

Reference

- Finding team of experts in a social network
  - [T. Lappas, K. Liu, E. Terzi KDD 2009]

evimaria@cs.bu.edu

Reference

- Finding team of experts in a social network
  - [T. Lappas, K. Liu, E. Terzi KDD 2009]

Reference

- Finding team of experts in a social network
  - [T. Lappas, K. Liu, E. Terzi KDD 2009]

Reference

- Finding team of experts in a social network
  - [T. Lappas, K. Liu, E. Terzi KDD 2009]

evimaria@cs.bu.edu

Reference

- Finding team of experts in a social network
  - [T. Lappas, K. Liu, E. Terzi KDD 2009]

> Expertise location in social networks: "How do I find an effective team of people that collectively can perform a given task"

evimaria@cs.bu.edu

Setting

- Experts (defining the set V, with |V|=n):
  - Every expert i is associated with **a set of skills** X_i
- Tasks
  - Every task T is associated with a set of skills (T) **required** for performing the task
- A social network of experts (G=(V,E))
  - Edges between experts indicate ability to work well together

	Team Formation
Experts' skills	Known
Participation of experts in teams	Unknown
Network structure	Known

evimaria@cs.bu.edu

Group-Formation Problem

Group-Formation Problem

- Given a task and a set of experts organized in a network find the subset of experts that can effectively perform the task

evimaria@cs.bu.edu

Group-Formation Problem

- Given a task and a set of experts organized in a network find the subset of experts that can effectively perform the task

evimaria@cs.bu.edu

Group-Formation Problem

- Given a task and a set of experts organized in a network find the subset of experts that can effectively perform the task
- Task: set of required skills

evimaria@cs.bu.edu

Group-Formation Problem

- Given a task and a set of experts organized in a network find the subset of experts that can effectively perform the task

- Task: set of required skills

evimaria@cs.bu.edu

Group-Formation Problem

- Given a task and a set of experts organized in a network find the subset of experts that can effectively perform the task

- Task: set of required skills

- Expert: has a set of skills

evimaria@cs.bu.edu

Group-Formation Problem

- Given a task and a set of experts organized in a network find the subset of experts that can effectively perform the task
- Task: set of required skills
- Expert: has a set of skills

evimaria@cs.bu.edu

Group-Formation Problem

- Given a task and a set of experts organized in a network find the subset of experts that can effectively perform the task
- Task: set of required skills
- Expert: has a set of skills
- Network: represents strength of relationships

Expertise networks

- Collaboration networks (e.g., DBLP graph, coauthor networks)
- Organizational structure of companies
- LinkedIn
- Geographical (map) of experts

evimaria@cs.bu.edu

What makes a team effective for a task?

- T = {algorithms, java, graphics, python}

Alice	Bob	Cynthia	David	Eleanor
{algorithms}	{python}	{graphics, java}	{graphics}	{graphics,java,python}

evimaria@cs.bu.edu

What makes a team effective for a task?

- T = {algorithms, java, graphics, python}

Alice {algorithms}	**B**ob {python}	**C**ynthia {graphics, java}	**D**avid {graphics}	**E**leanor {graphics,java,python}

Alice {algorithms}	**E**leanor {graphics,java,python}

What makes a team effective for a task?

- T = {algorithms, java, graphics, python}

Alice
{algorithms}

Bob
{python}

Cynthia
{graphics, java}

David
{graphics}

Eleanor
{graphics,java,python}

Alice
{algorithms}

Eleanor
{graphics,java,python}

Coverage: For every required skill in T there is at least one team member that has it

Problem definition – v.0

- Given a task and a set of individuals, find the subset (team) of individuals that can perform the given task.

evimaria@cs.bu.edu

Is coverage enough?

T={algorithms,java,graphics,python}

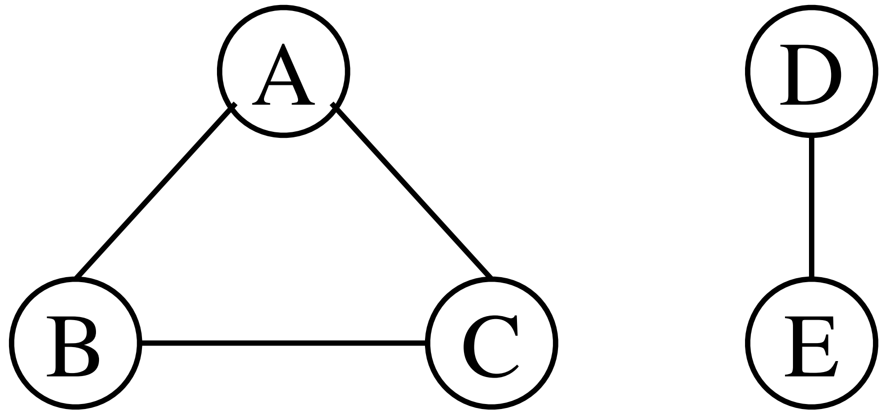
Alice	Bob	Cynthia	David	Eleanor
{algorithms}	{python}	{graphics, java}	{graphics}	{graphics,java,python}

evimaria@cs.bu.edu

Is coverage enough?

T={algorithms,java,graphics,python}

Alice	Bob	Cynthia	David	Eleanor
{algorithms}	{python}	{graphics, java}	{graphics}	{graphics,java,python}

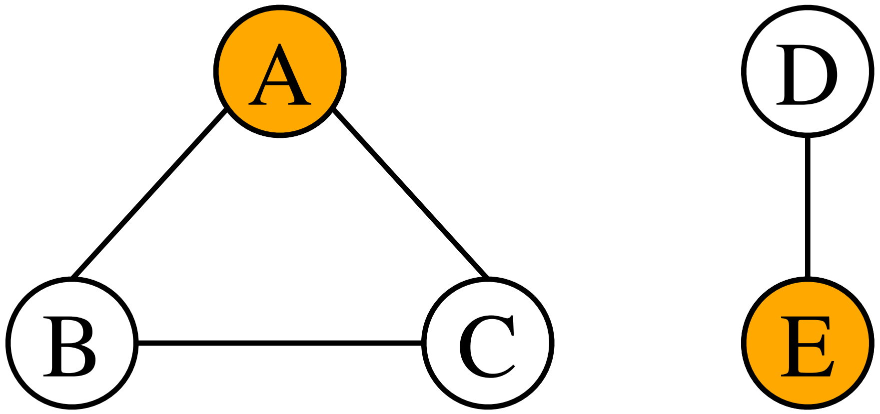


evimaria@cs.bu.edu

Is coverage enough?

T={algorithms,java,graphics,python}

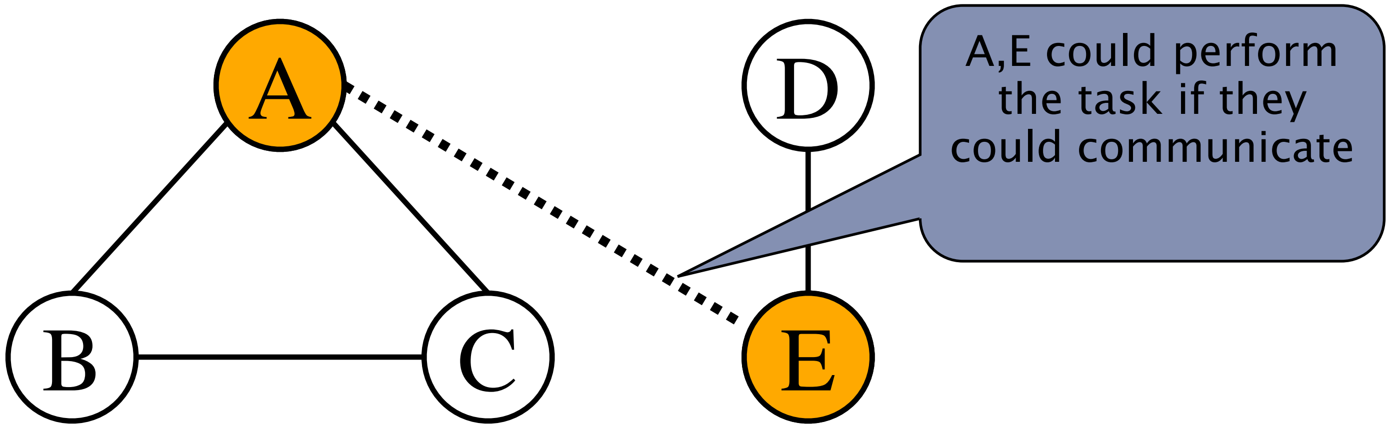
Alice	Bob	Cynthia	David	Eleanor
{algorithms}	{python}	{graphics, java}	{graphics}	{graphics,java,python}



Is coverage enough?

T={algorithms,java,graphics,python}

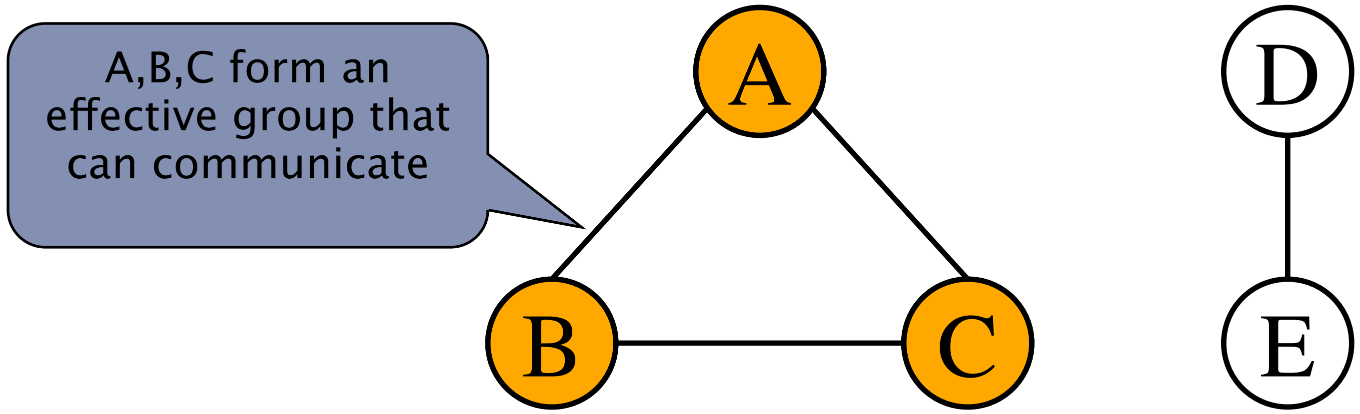
Alice	Bob	Cynthia	David	Eleanor
{algorithms}	{python}	{graphics, java}	{graphics}	{graphics,java,python}



Is coverage enough?

T={algorithms,java,graphics,python}

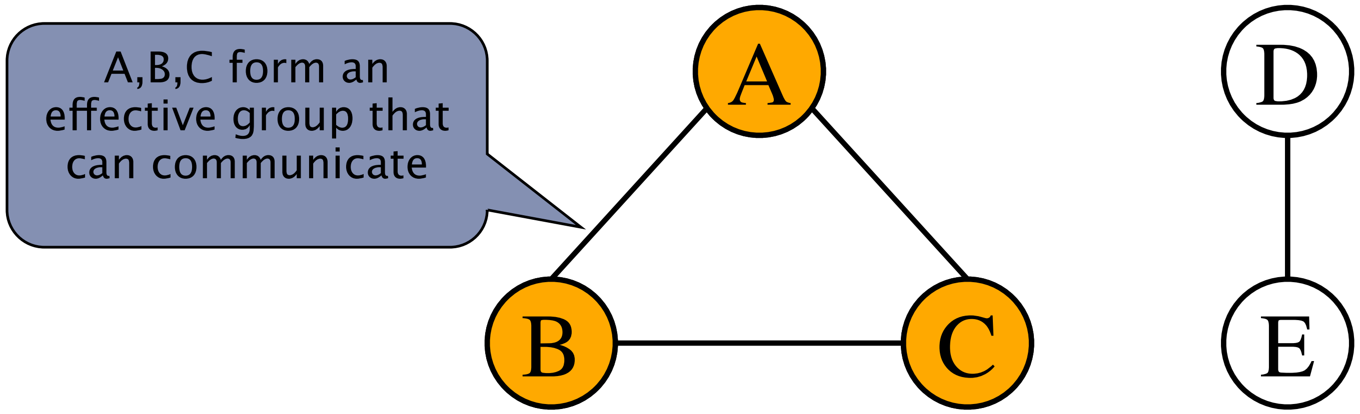
Alice	Bob	Cynthia	David	Eleanor
{algorithms}	{python}	{graphics, java}	{graphics}	{graphics,java,python}



Is coverage enough?

T={algorithms,java,graphics,python}

Alice	Bob	Cynthia	David	Eleanor
{algorithms}	{python}	{graphics, java}	{graphics}	{graphics,java,python}



Communication: the members of the team must be able to efficiently communicate and work together

Problem definition – v.1

evimaria@cs.bu.edu

Problem definition – v.1

- Given a task and a social network of individuals, find the subset (team) of individuals that can effectively perform the given task.

evimaria@cs.bu.edu

Problem definition – v.1

- Given a task and a social network of individuals, find the subset (team) of individuals that can effectively perform the given task.

evimaria@cs.bu.edu

Problem definition – v.1

- Given a task and a social network of individuals, find the subset (team) of individuals that can effectively perform the given task.

evimaria@cs.bu.edu

Problem definition – v.1

- Given a task and a social network of individuals, find the subset (team) of individuals that can effectively perform the given task.

- **Thesis:** Good teams are teams that have the necessary skills and can also communicate effectively

evimaria@cs.bu.edu

How to measure effective communication?

- Diameter of the subgraph defined by the group members

evimaria@cs.bu.edu

How to measure effective communication?

The longest shortest path between any two nodes in the subgraph

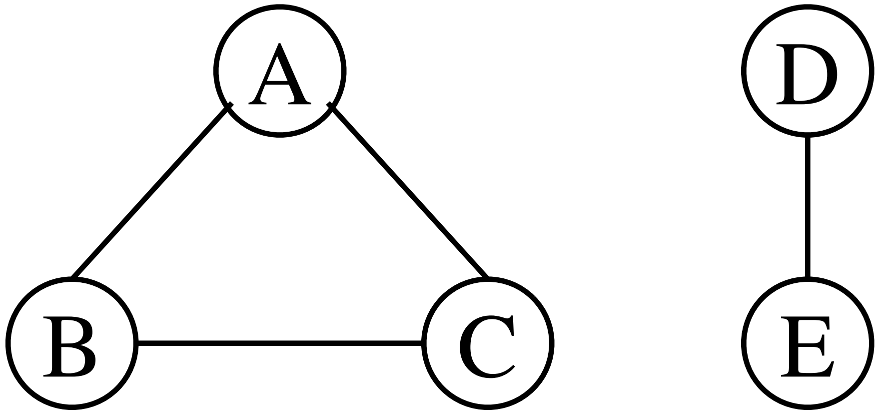
- Diameter of the subgraph defined by the group members

evimaria@cs.bu.edu

How to measure effective communication?

The longest shortest path between any two nodes in the subgraph

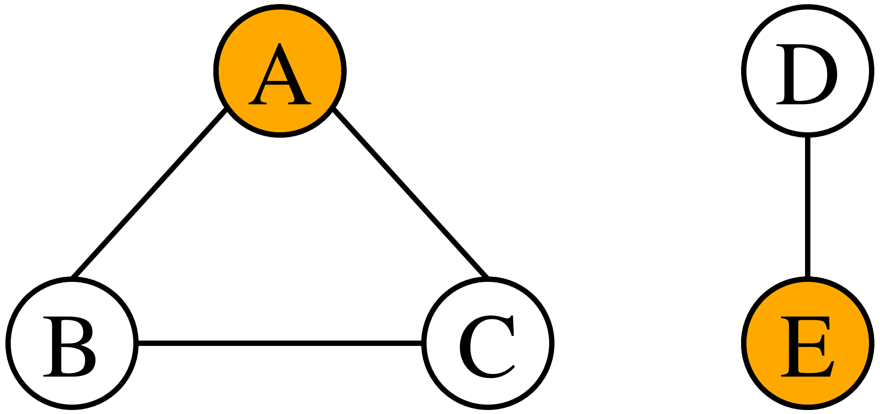
- Diameter of the subgraph defined by the group members



How to measure effective communication?

The longest shortest path between any two nodes in the subgraph

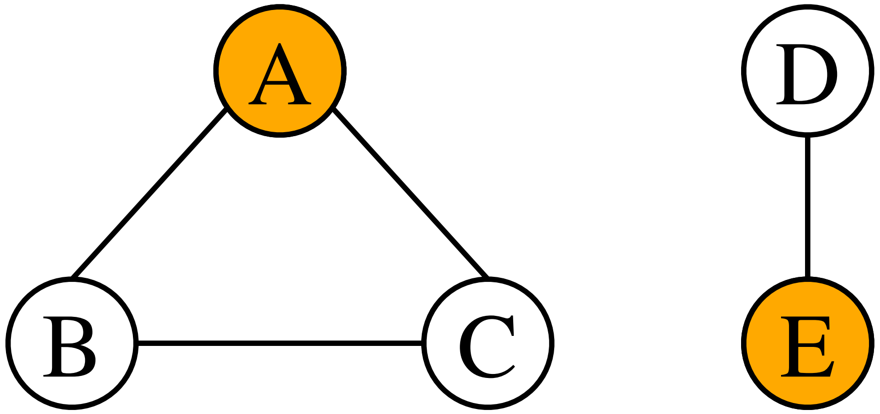
- Diameter of the subgraph defined by the group members



How to measure effective communication?

The longest shortest path between any two nodes in the subgraph

- Diameter of the subgraph defined by the group members



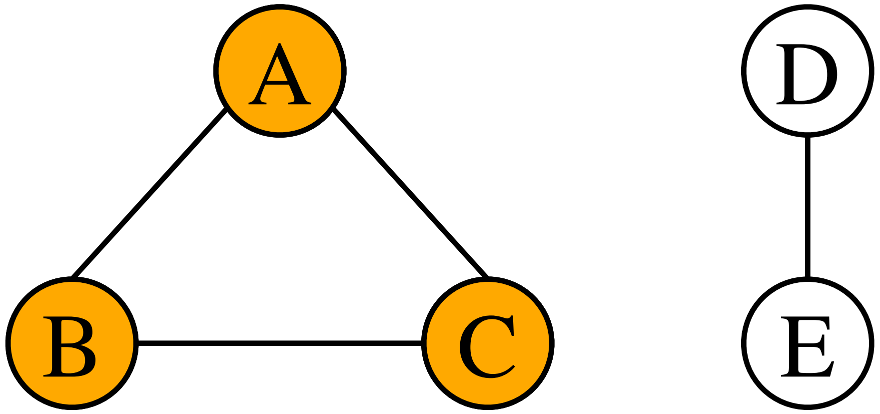
diameter = infty

BOSTON UNIVERSITY
evimaria@cs.bu.edu

How to measure effective communication?

The longest shortest path between any two nodes in the subgraph

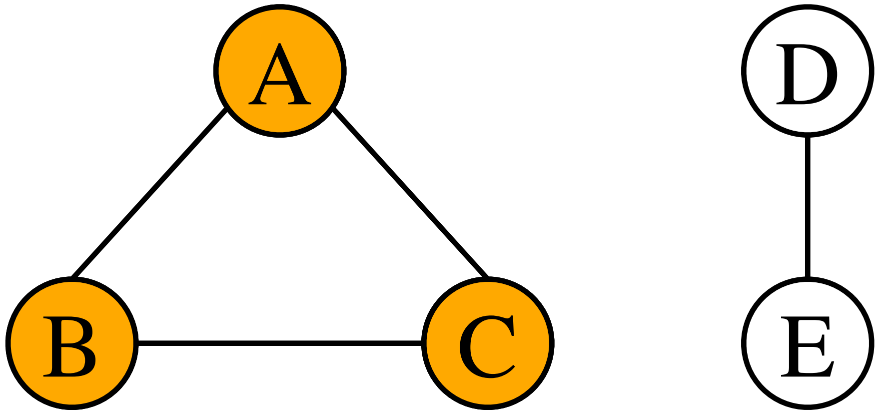
- Diameter of the subgraph defined by the group members



How to measure effective communication?

The longest shortest path between any two nodes in the subgraph

- Diameter of the subgraph defined by the group members



diameter = 1

evimaria@cs.bu.edu

How to measure effective communication?

- MST (Minimum spanning tree) of the subgraph defined by the group members

evimaria@cs.bu.edu

How to measure effective communication?

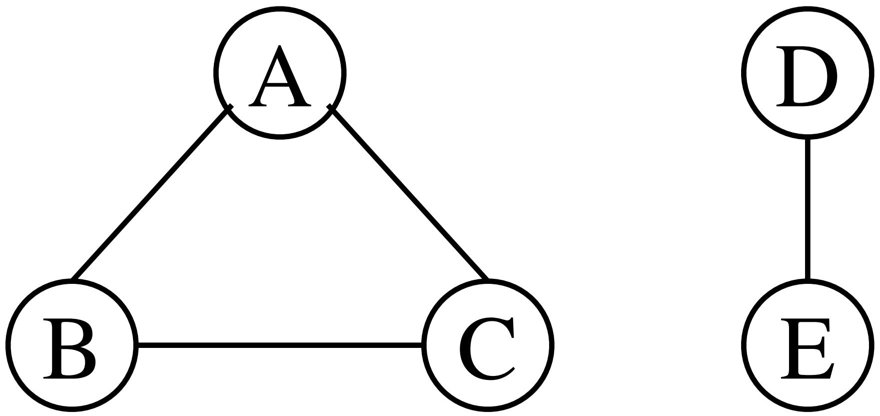
The total weight of the edges of a tree that spans all the team nodes

- MST (Minimum spanning tree) of the subgraph defined by the group members

How to measure effective communication?

The total weight of the edges of a tree that spans all the team nodes

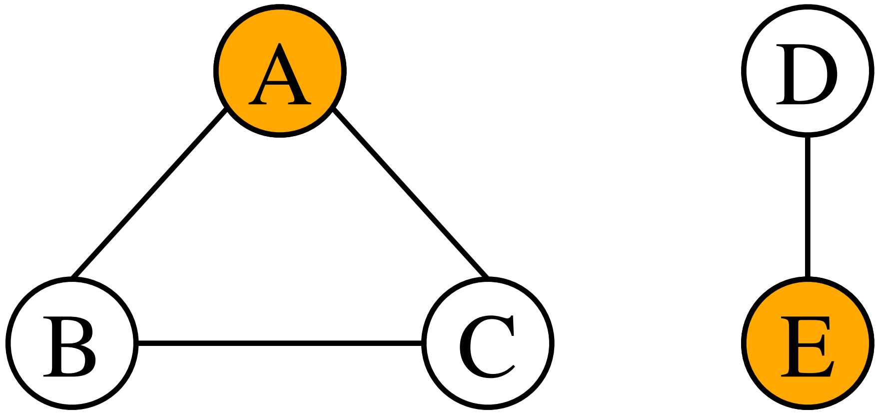
- MST (Minimum spanning tree) of the subgraph defined by the group members



How to measure effective communication?

The total weight of the edges of a tree that spans all the team nodes

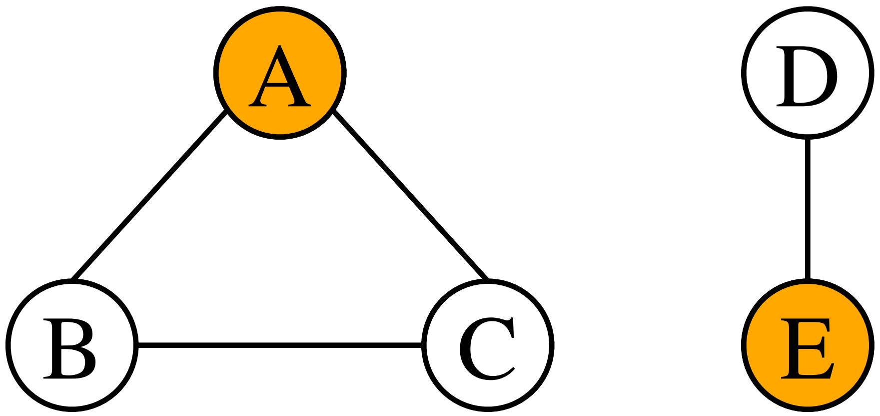
- MST (Minimum spanning tree) of the subgraph defined by the group members



How to measure effective communication?

The total weight of the edges of a tree that spans all the team nodes

- MST (Minimum spanning tree) of the subgraph defined by the group members



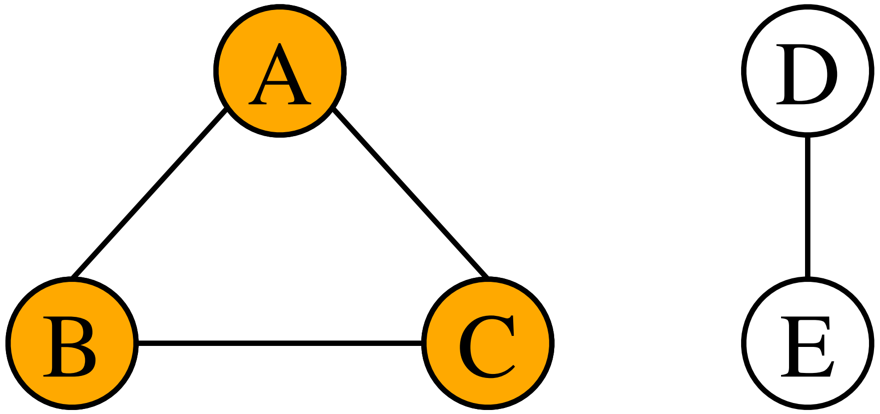
MST = infty

evimaria@cs.bu.edu

How to measure effective communication?

The total weight of the edges of a tree that spans all the team nodes

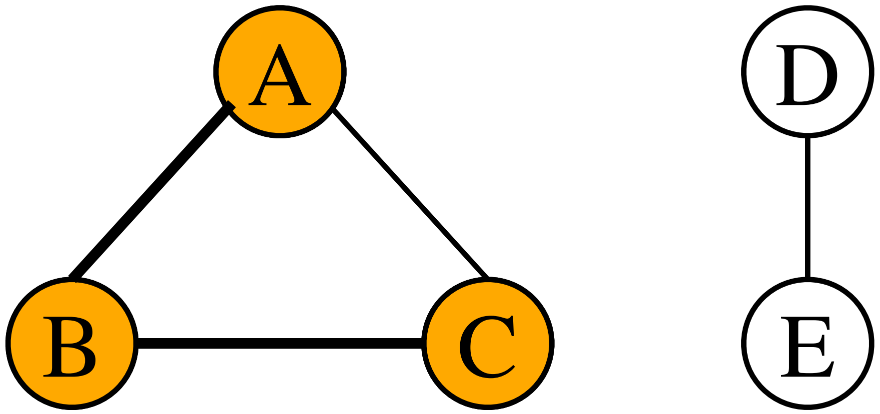
- MST (Minimum spanning tree) of the subgraph defined by the group members



How to measure effective communication?

The total weight of the edges of a tree that spans all the team nodes

- MST (Minimum spanning tree) of the subgraph defined by the group members



MST = 2

Problem definition – v.1.1

Problem definition – v.1.1

- Given a task and a social network G of experts, find the subset (team) of experts that can perform the given task and they define a subgraph in G with the minimum diameter.

- Problem is NP-hard

evimaria@cs.bu.edu

Algorithms for minimizing the diameter : RarestFirst

- Find Rarest skill α_{rare} required for a task
- S_{rare} group of people that have α_{rare}
- Evaluate star graphs, centered at individuals from S_{rare}
- Report cheapest star

evimaria@cs.bu.edu

Algorithms for minimizing the diameter : RarestFirst

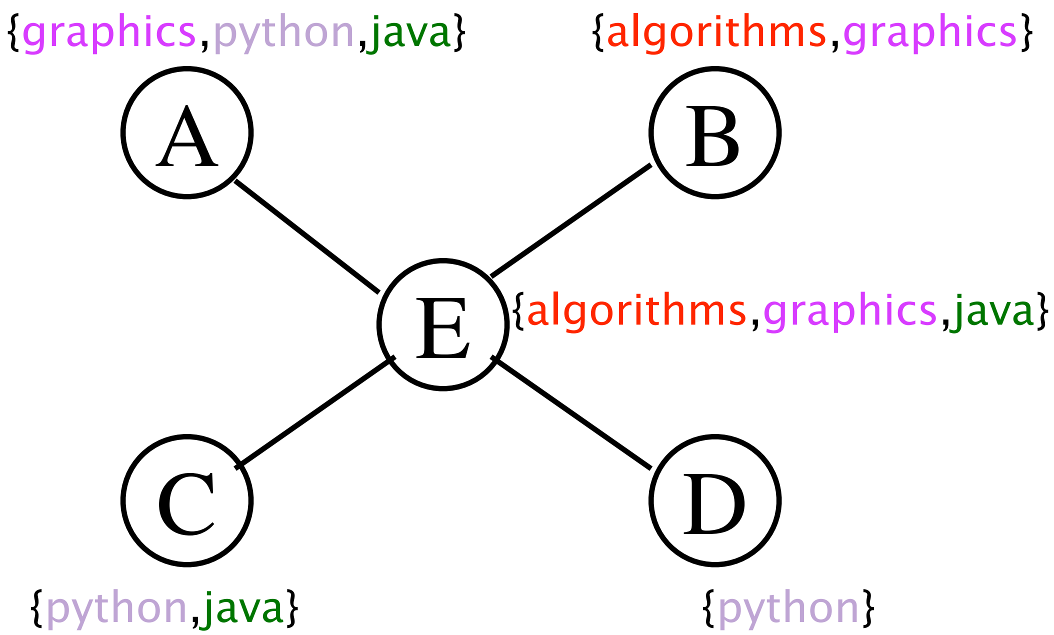
- Find Rarest skill α_{rare} required for a task
- S_{rare} group of people that have α_{rare}
- Evaluate star graphs, centered at individuals from S_{rare}
- Report cheapest star

Running time: Quadratic to the number of nodes

evimaria@cs.bu.edu

The RarestFirst algorithm

T={algorithms,java,graphics,python}



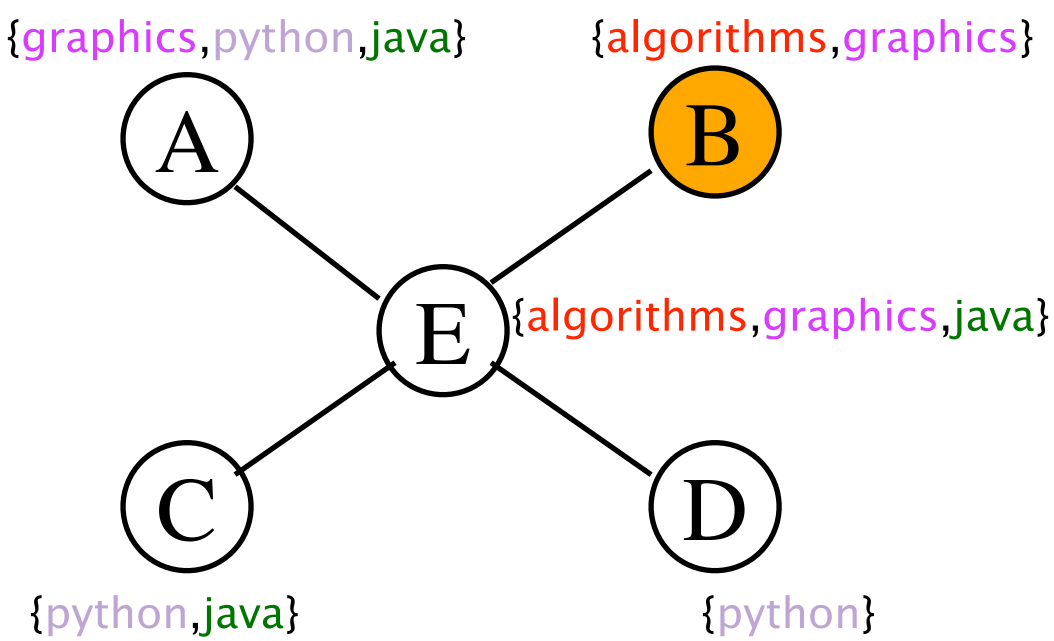
α_{rare}= algorithms

S_{rare}={Bob, Eleanor}

evimaria@cs.bu.edu

The RarestFirst algorithm

T={algorithms,java,graphics,python}

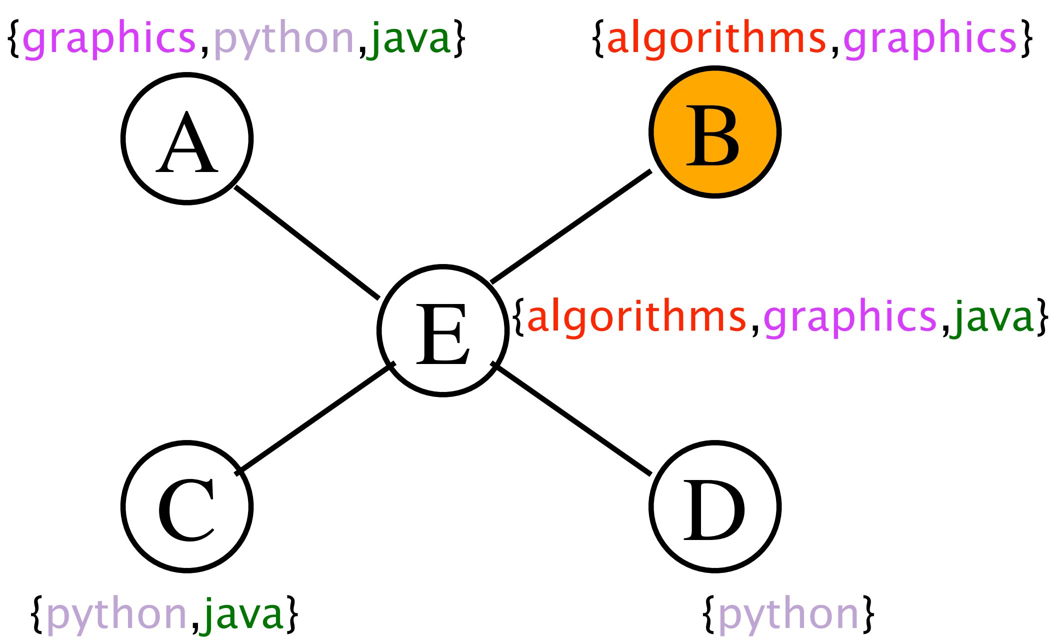


α_{rare}= algorithms

S_{rare}={Bob, Eleanor}

The RarestFirst algorithm

$T = \{$algorithms, java, graphics, python$\}$



Skills:

algorithms

graphics

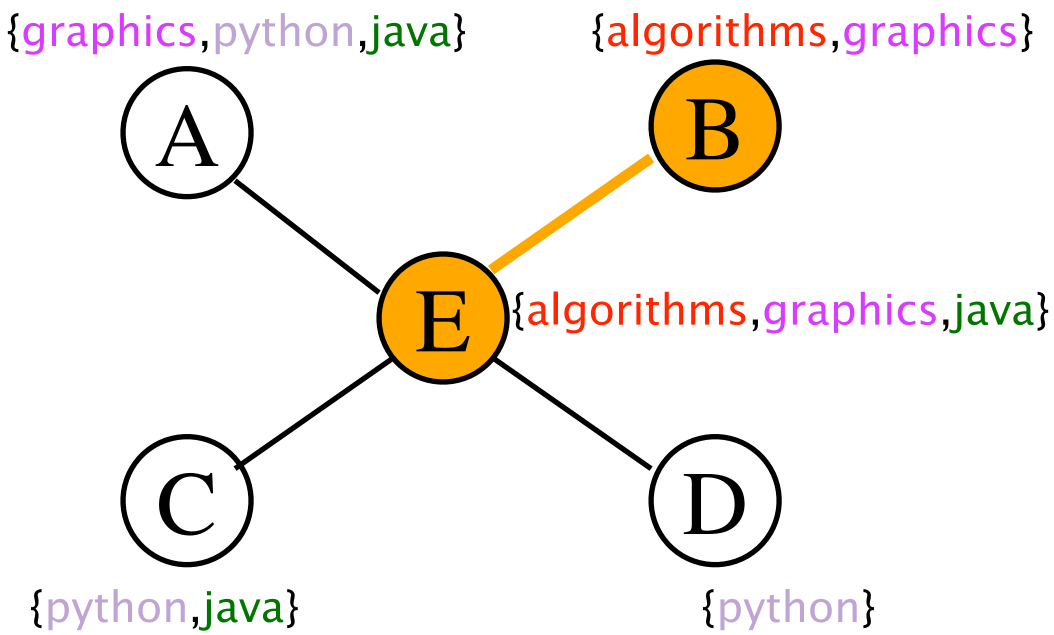
α_{rare} = algorithms

$S_{rare} = \{B_{ob}, E_{leanor}\}$

evimaria@cs.bu.edu

The RarestFirst algorithm

T={algorithms,java,graphics,python}



Skills:

algorithms

graphics

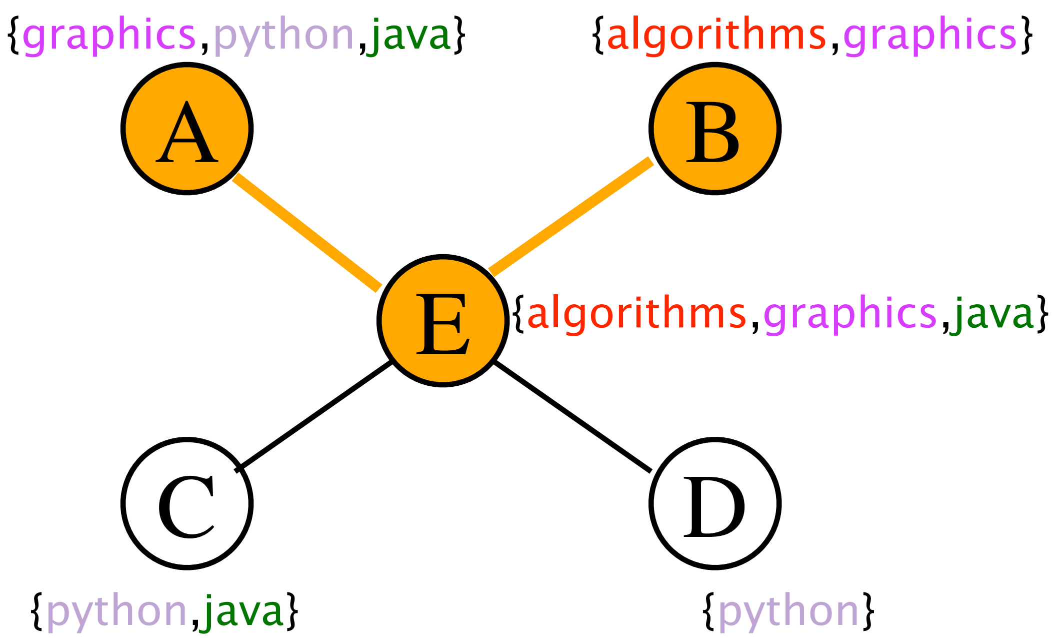
java

α_{rare}= algorithms

S_{rare}={Bob, Eleanor}

evimaria@cs.bu.edu

The RarestFirst algorithm



T={algorithms,java,graphics,python}

Skills:

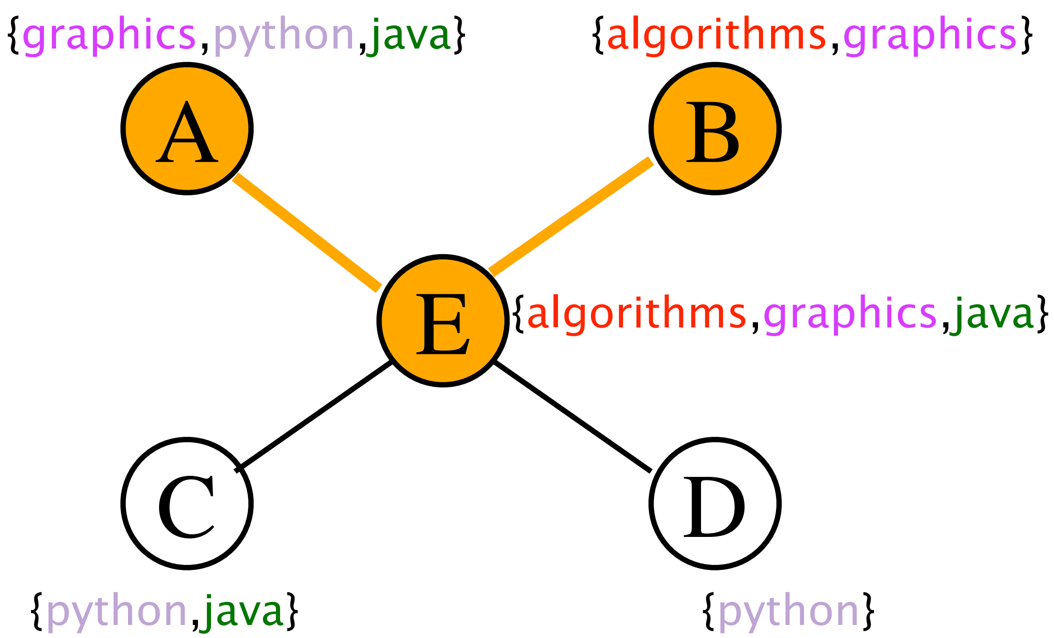
- algorithms
- graphics
- java
- python

α_{rare}= algorithms

S_{rare}={B~ob~, E~leanor~}

The RarestFirst algorithm

$T = \{$algorithms, java, graphics, python$\}$



Skills:

algorithms

graphics

java

python

α_{rare} = algorithms

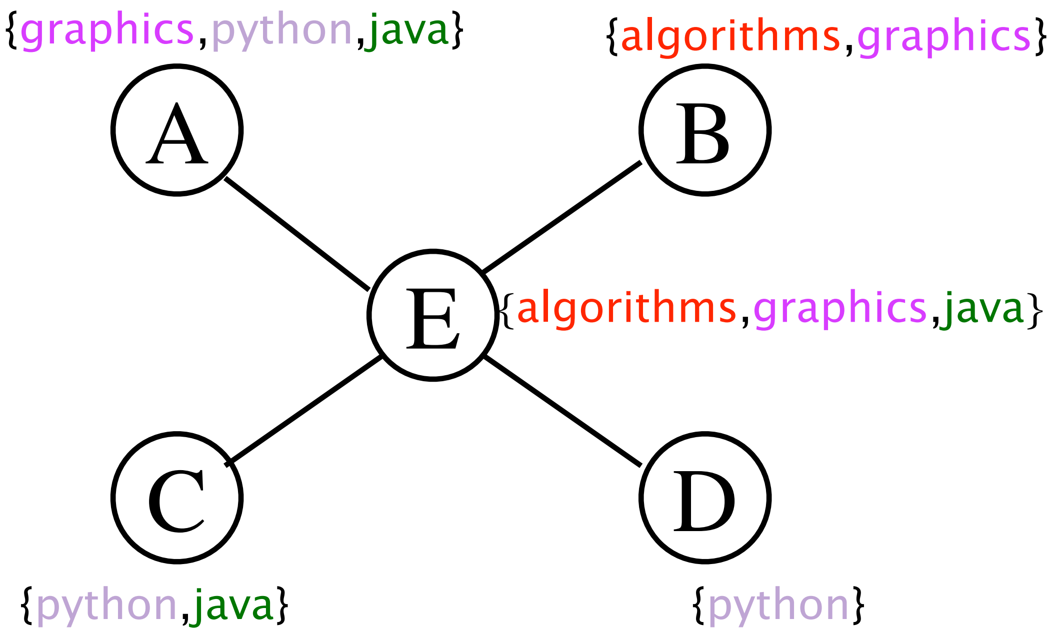
S_{rare} = {B_{ob}, E_{leanor}}

Diameter = 2

evimaria@cs.bu.edu

The RarestFirst algorithm

T={algorithms,java,graphics,python}

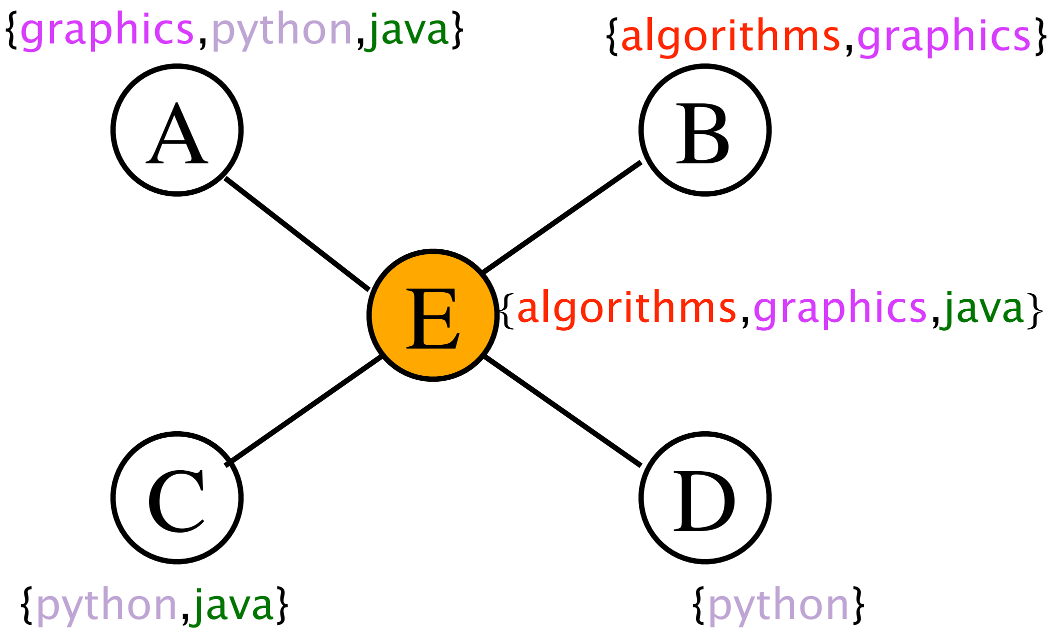


α_{rare}= algorithms

S_{rare}={B_{ob}, E_{leanor}}

The RarestFirst algorithm

T={algorithms,java,graphics,python}



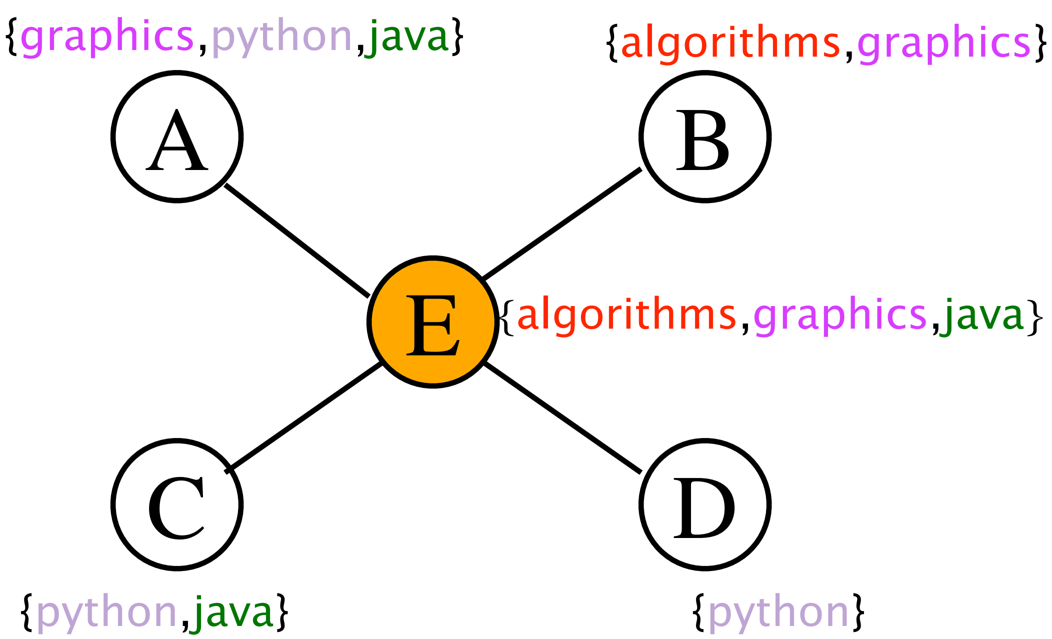
α_{rare} = algorithms

S_{rare}={B_{ob}, E_{leanor}}

evimaria@cs.bu.edu

The RarestFirst algorithm

T={algorithms,java,graphics,python}



Skills:

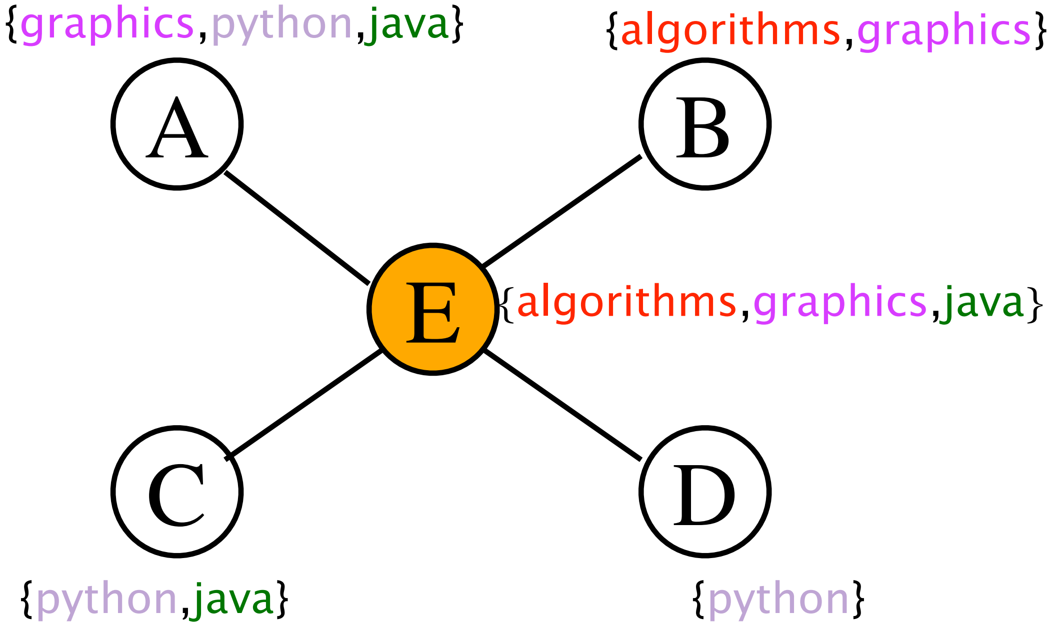
α_{rare}= algorithms

S_{rare}={Bob, Eleanor}

evimaria@cs.bu.edu

The RarestFirst algorithm

$T=\{$algorithms, java, graphics, python$\}$



Skills:

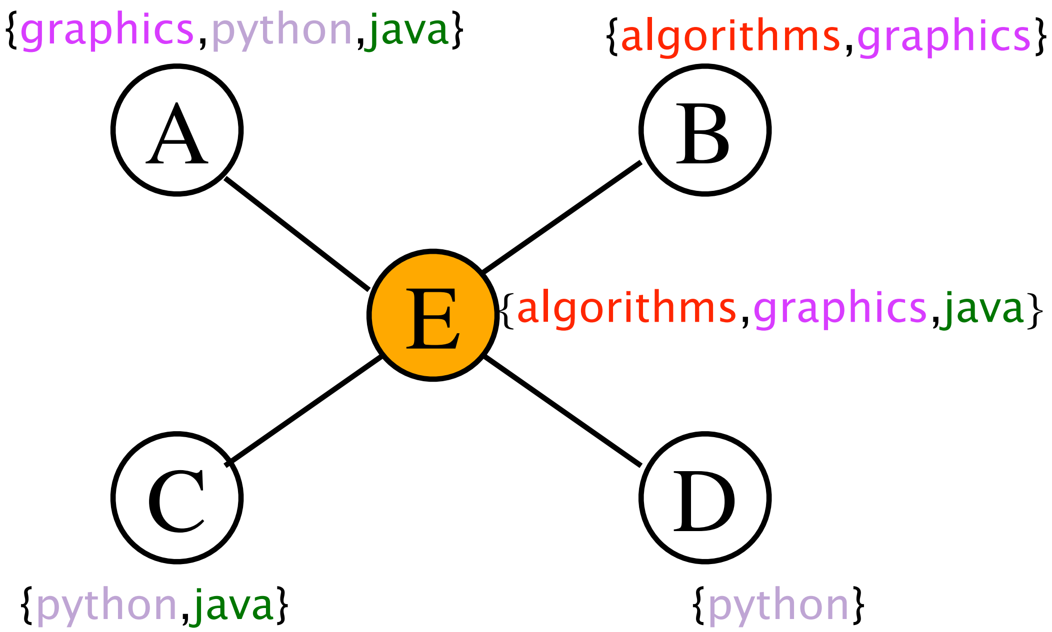
algorithms

$\alpha_{rare}=$ algorithms

$S_{rare}=\{B_{ob},\ E_{leanor}\}$

The RarestFirst algorithm

$T=\{$algorithms, java, graphics, python$\}$



Skills:

algorithms

graphics

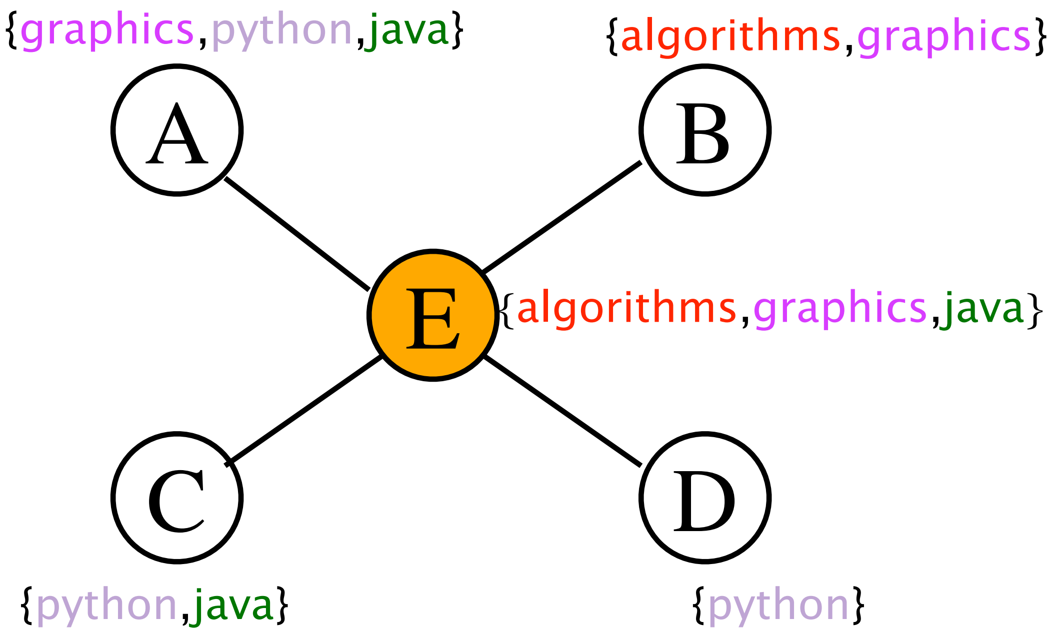
α_{rare} = algorithms

$S_{rare}=\{B_{ob},\ E_{leanor}\}$

evimaria@cs.bu.edu

The RarestFirst algorithm

T={algorithms,java,graphics,python}



Skills:

algorithms

graphics

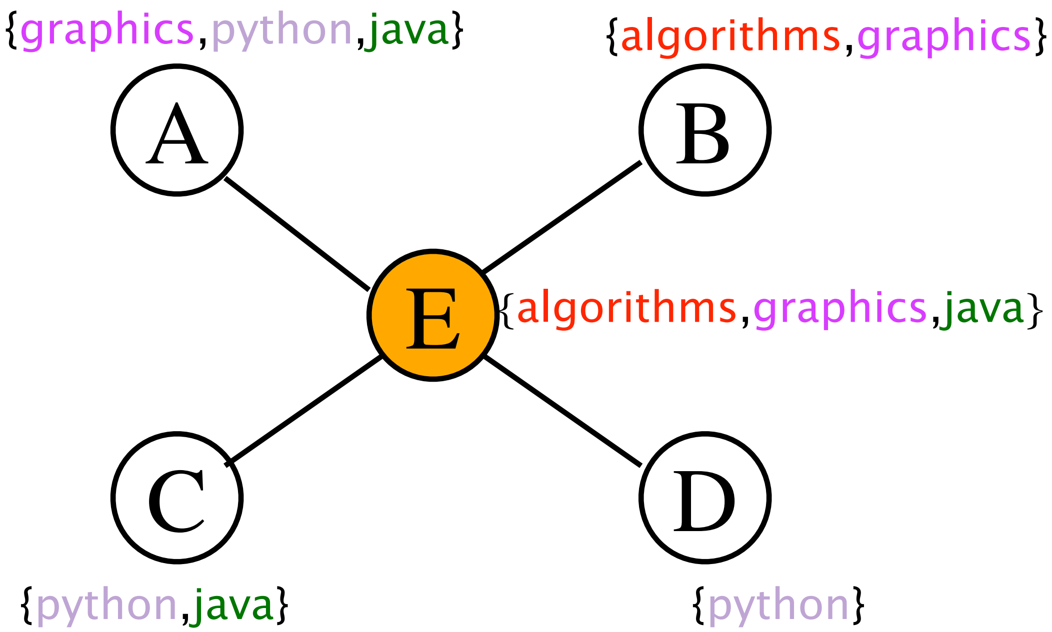
java

α_{rare} = algorithms

S_{rare}={Bob, Eleanor}

The RarestFirst algorithm

$T=\{$algorithms, java, graphics, python$\}$



Skills:

algorithms

graphics

java

python

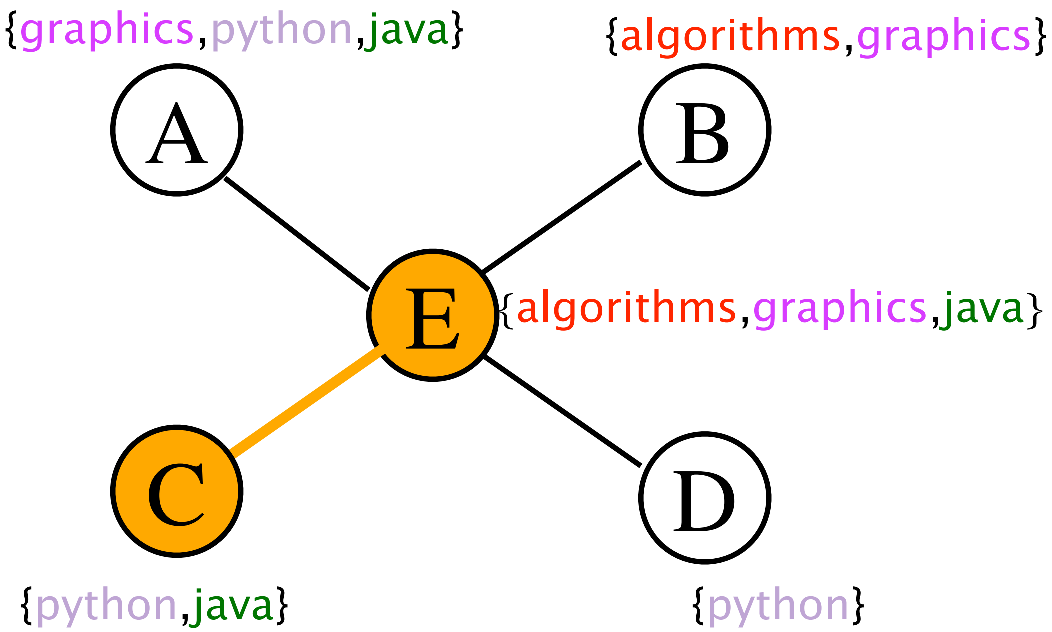
$\alpha_{rare}=$ algorithms

$S_{rare}=\{B_{ob}, E_{leanor}\}$

evimaria@cs.bu.edu

The RarestFirst algorithm

$T=\{$algorithms, java, graphics, python$\}$



Skills:

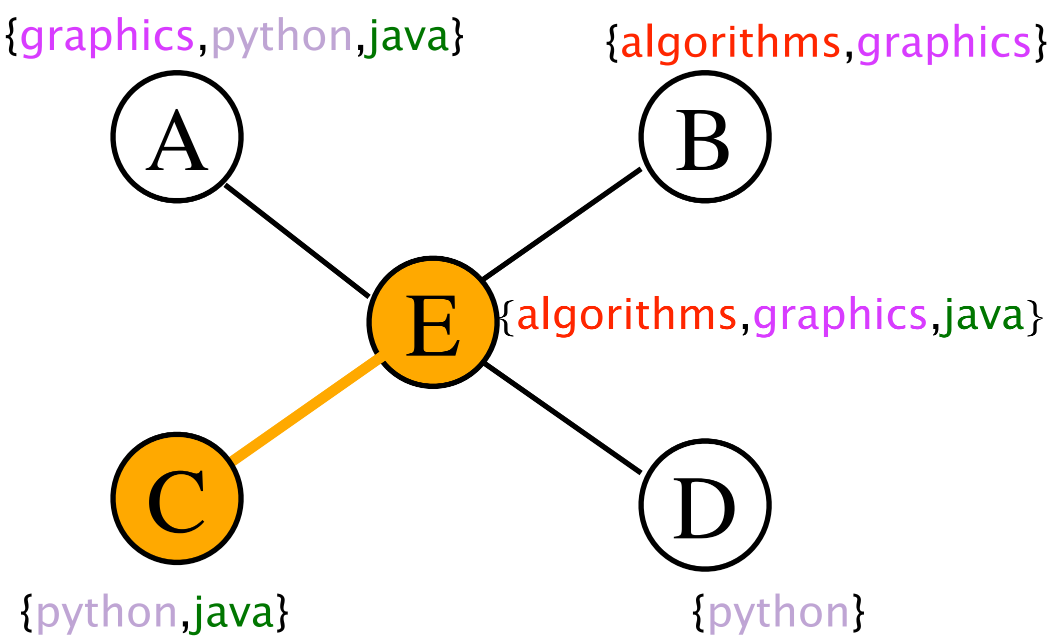
- algorithms
- graphics
- java
- python

α_{rare} = algorithms

S_{rare} = {Bob, Eleanor}

The RarestFirst algorithm

T={algorithms,java,graphics,python}



Skills:

algorithms

graphics

java

python

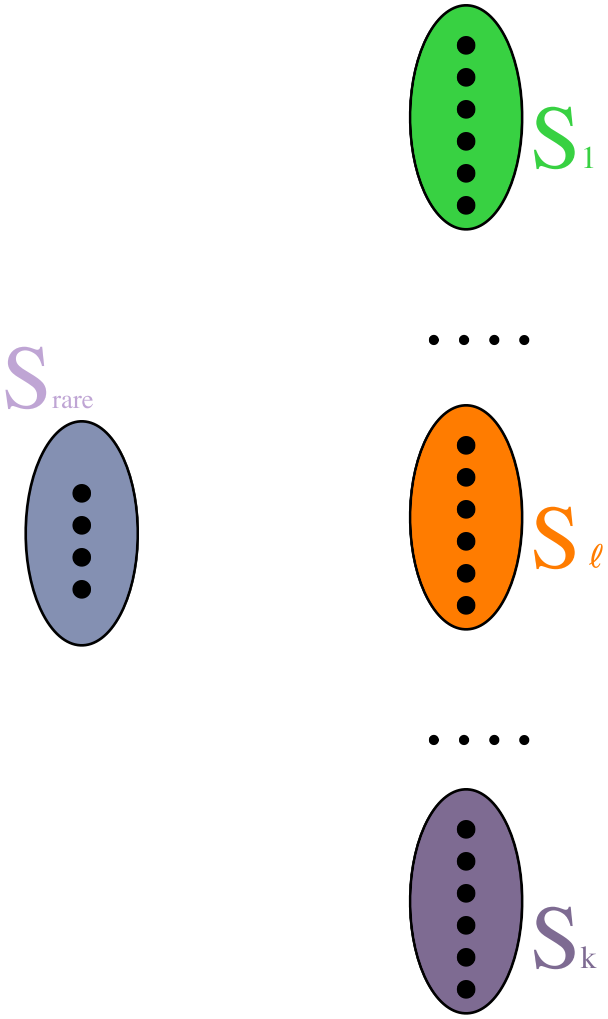
α_{rare} = algorithms

S_{rare}={B_{ob}, E_{leanor}}

Diameter = 1

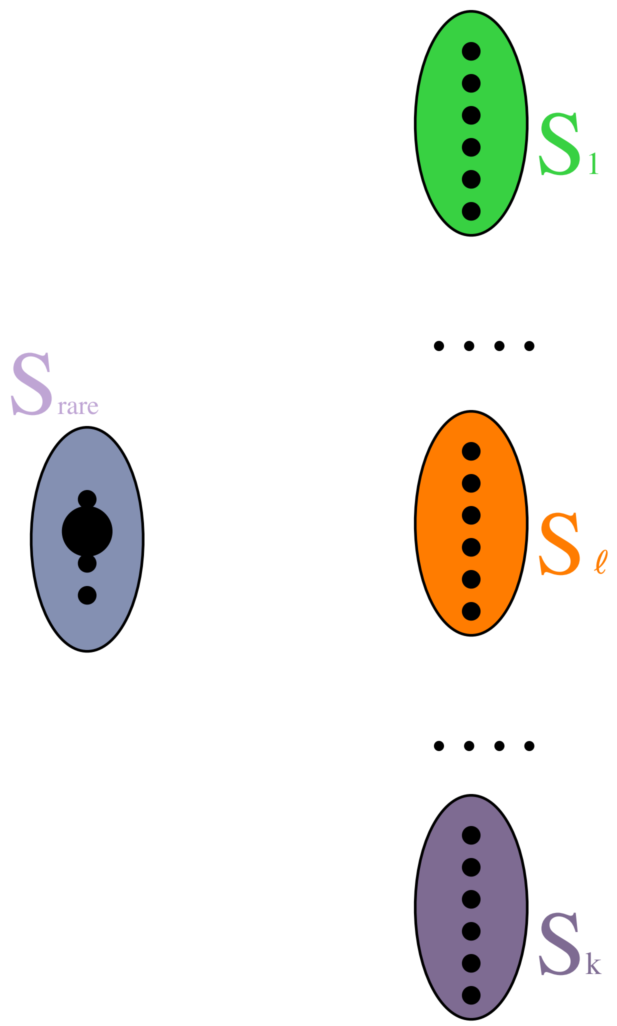
evimaria@cs.bu.edu

Analysis of the RarestFirst algorithm (metric graphs)

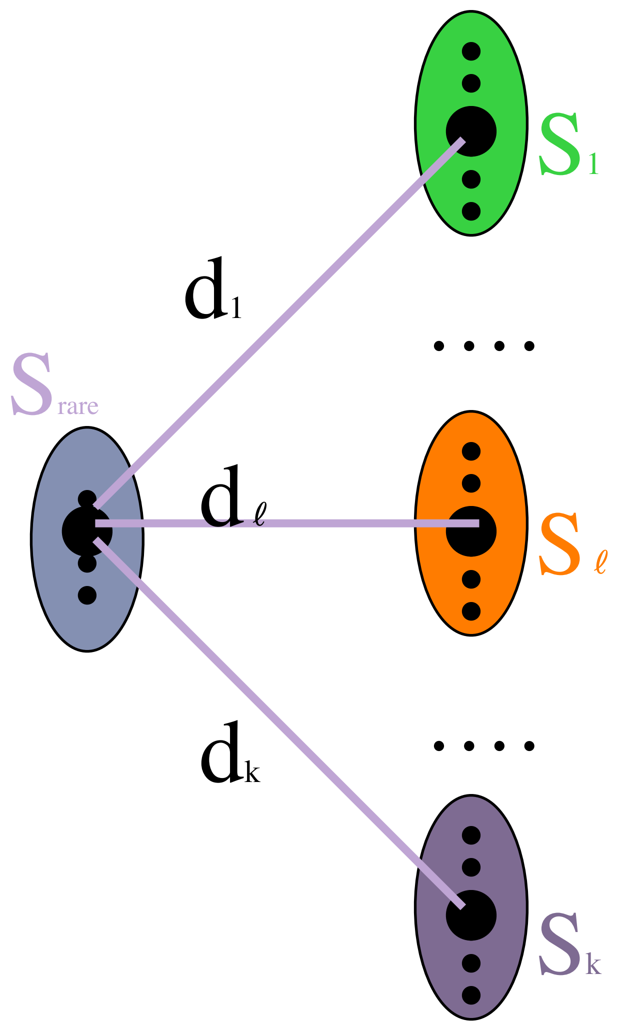


evimaria@cs.bu.edu

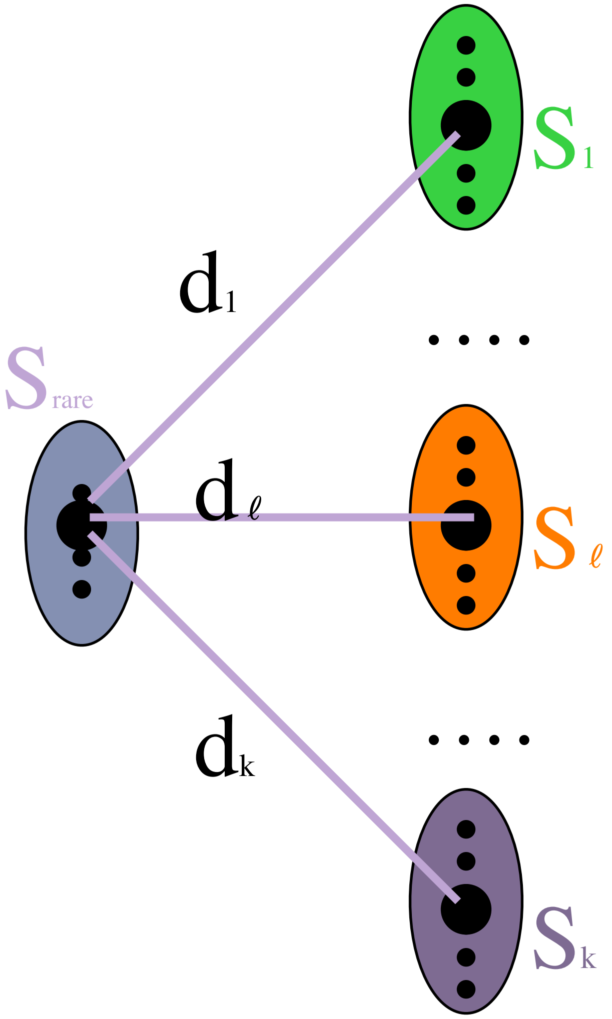
Analysis of the RarestFirst algorithm (metric graphs)



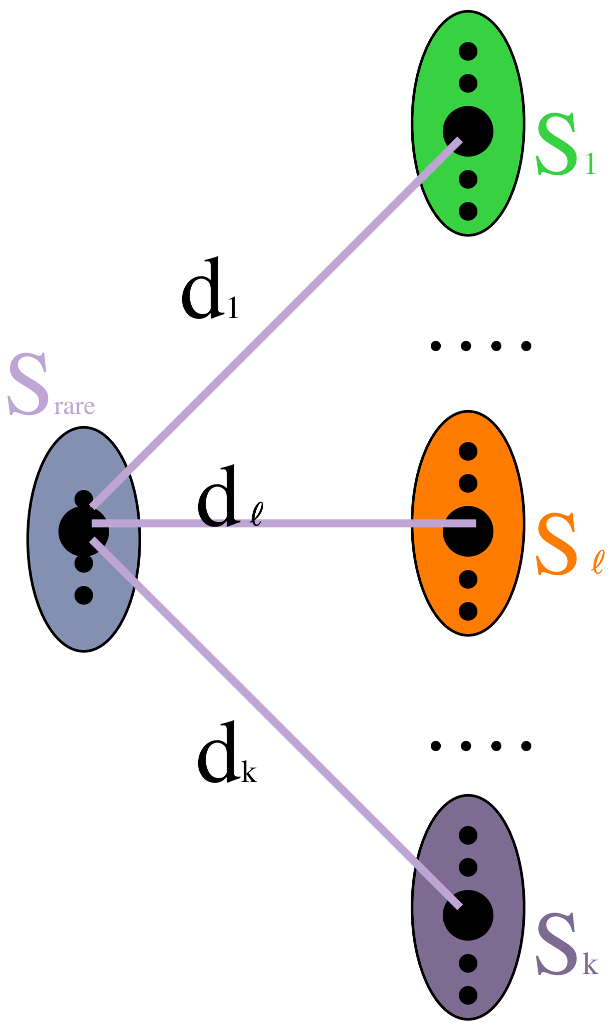
Analysis of the RarestFirst algorithm (metric graphs)



Analysis of the RarestFirst algorithm (metric graphs)

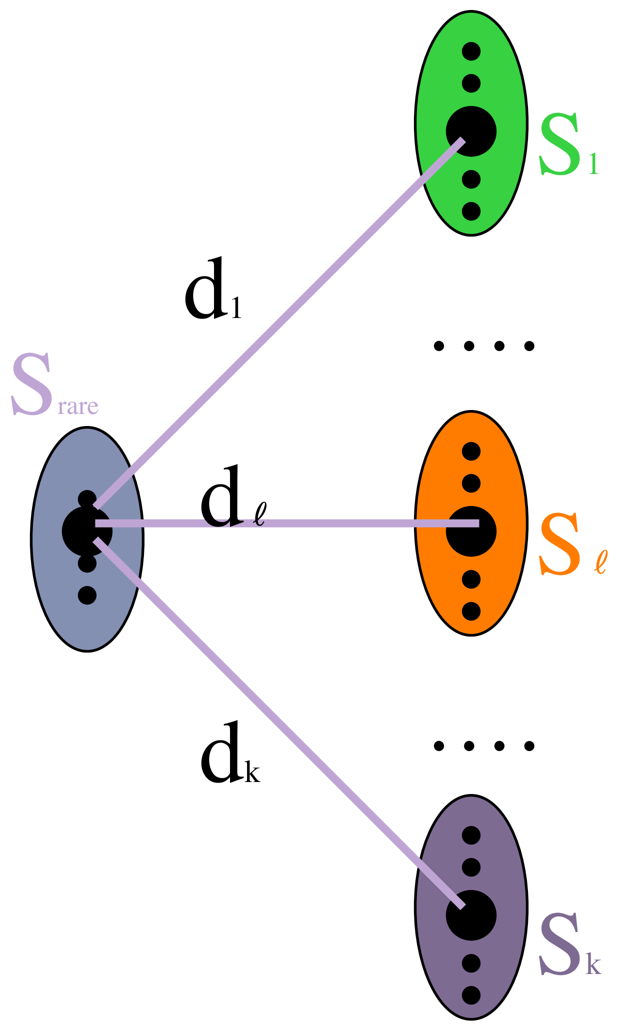


Analysis of the RarestFirst algorithm (metric graphs)



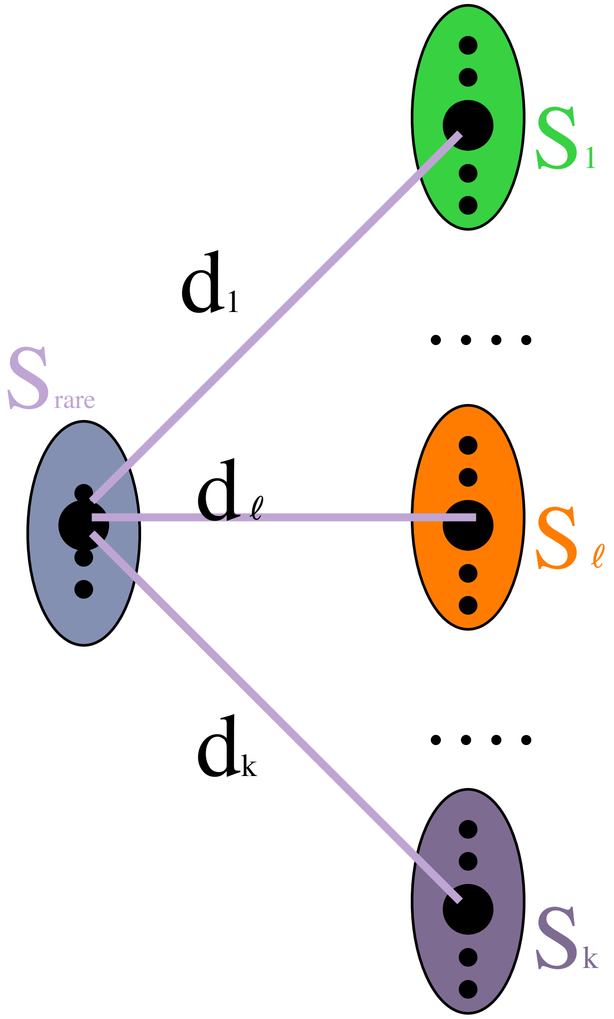
- $D = \max\{d_{\ell}, d_k, d_{\ell k}\}$

Analysis of the RarestFirst algorithm (metric graphs)



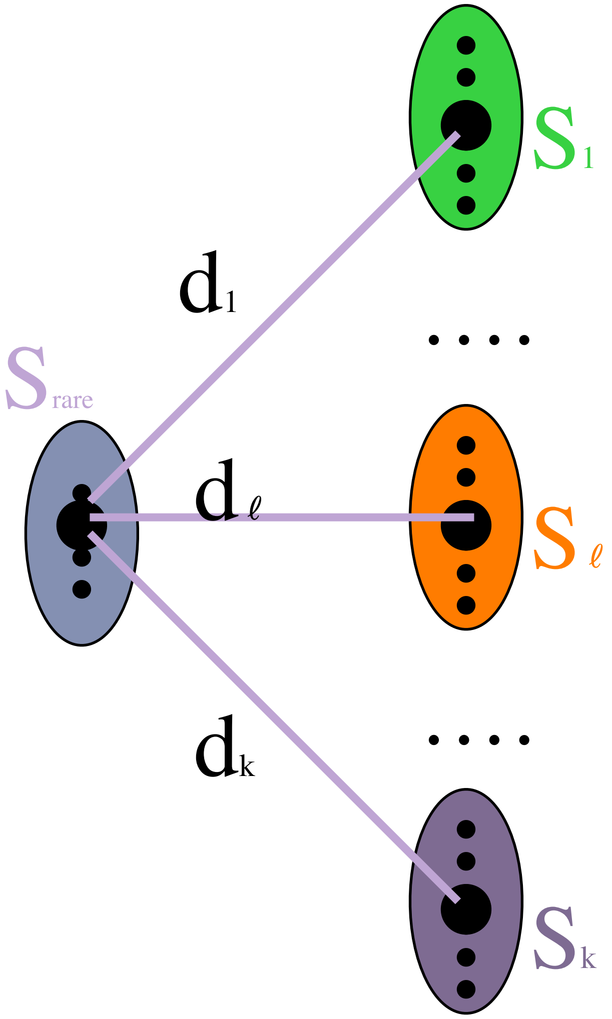
- $D = \max\{d_\ell, d_k, d_{\ell k}\}$

Analysis of the RarestFirst algorithm (metric graphs)



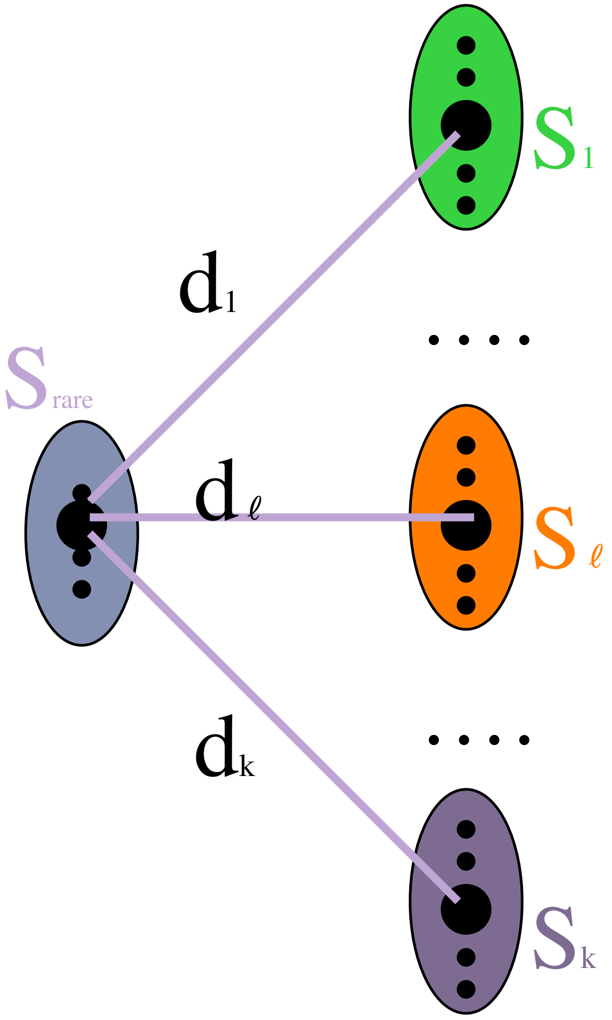
- $D = \max\{d_\ell, d_k, d_{\ell k}\}$
- Fact: $OPT \geq d_\ell$

Analysis of the RarestFirst algorithm (metric graphs)



- $D = \max\{d_\ell, d_k, d_{\ell k}\}$
- Fact: $OPT \geq d_\ell$

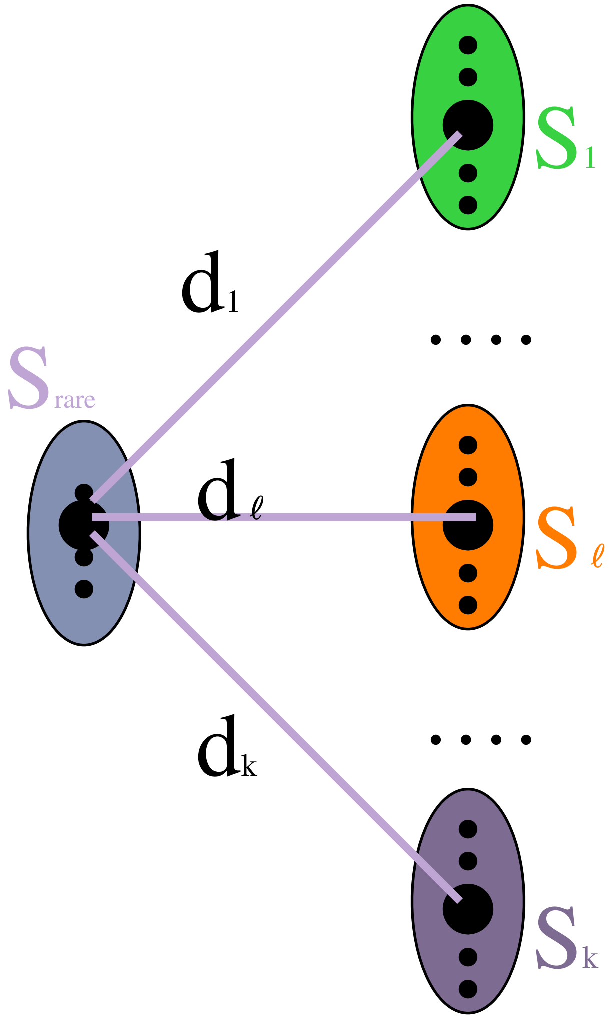
Analysis of the RarestFirst algorithm (metric graphs)



- $D = \max\{d_\ell, d_k, d_{\ell k}\}$
- Fact: OPT $\geq d_\ell$
- Fact: OPT $\geq d_k$

evimaria@cs.bu.edu

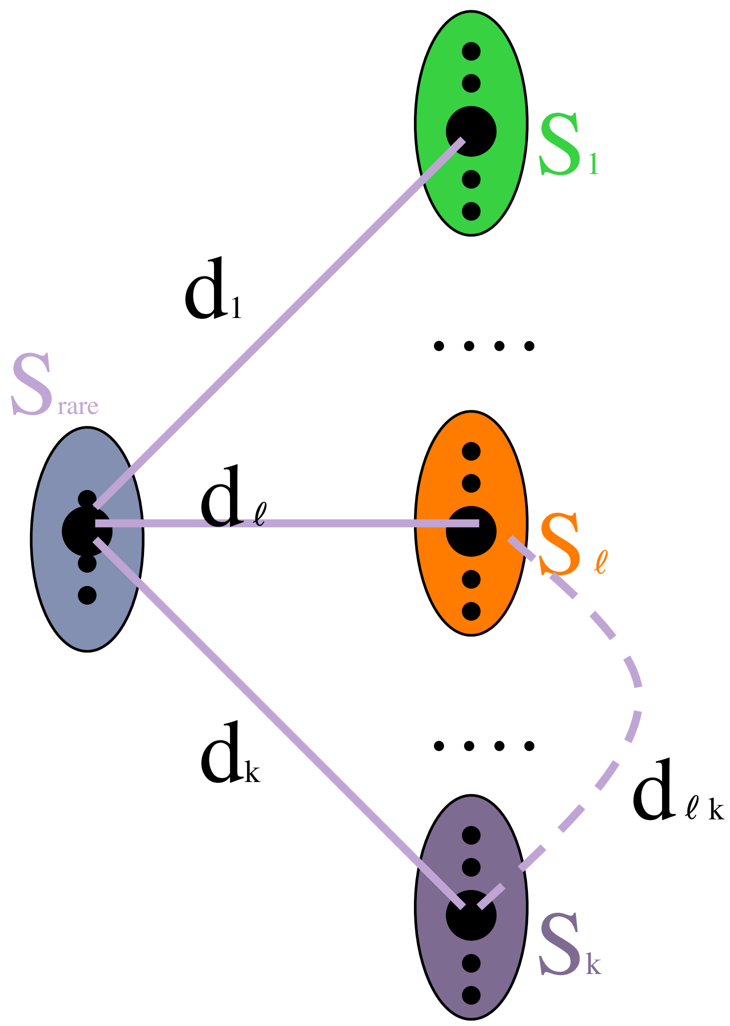
Analysis of the RarestFirst algorithm (metric graphs)



- $D = \max \{d_\ell, d_k, d_{\ell k}\}$
- Fact: $OPT \geq d_\ell$
- Fact: $OPT \geq d_k$

evimaria@cs.bu.edu

Analysis of the RarestFirst algorithm (metric graphs)



- $D = \max\{d_{\ell}, d_k, d_{\ell k}\}$
- Fact: $OPT \geq d_{\ell}$
- Fact: $OPT \geq d_k$
- $D \leq d_{\ell k} \leq d_{\ell} + d_k \leq 2*OPT$

Problem definition – v.1.2

Problem definition – v.1.2

- Given a task and a social network G of experts, find the subset (team) of experts that can perform the given task and they define a subgraph in G with the minimum MST cost.

- Problem is NP-hard

evimaria@cs.bu.edu

The SteinerTree problem

- Graph G=(V,E)

Required vertices

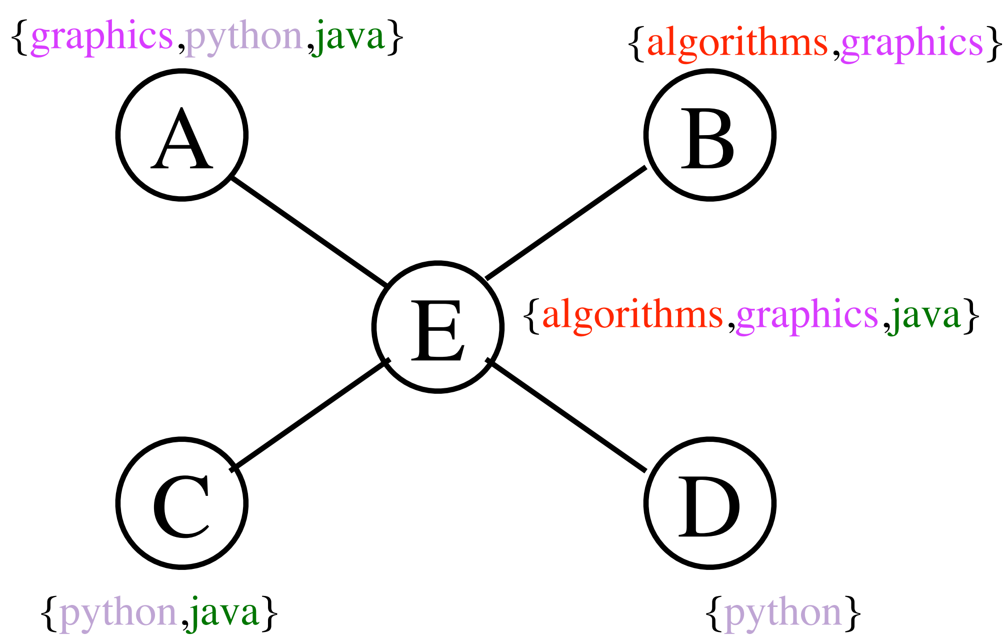
- Partition of V into V = {R,N}

- Find G' subgraph of G such that G' contains all the required vertices (R) and MST(G') is minimized

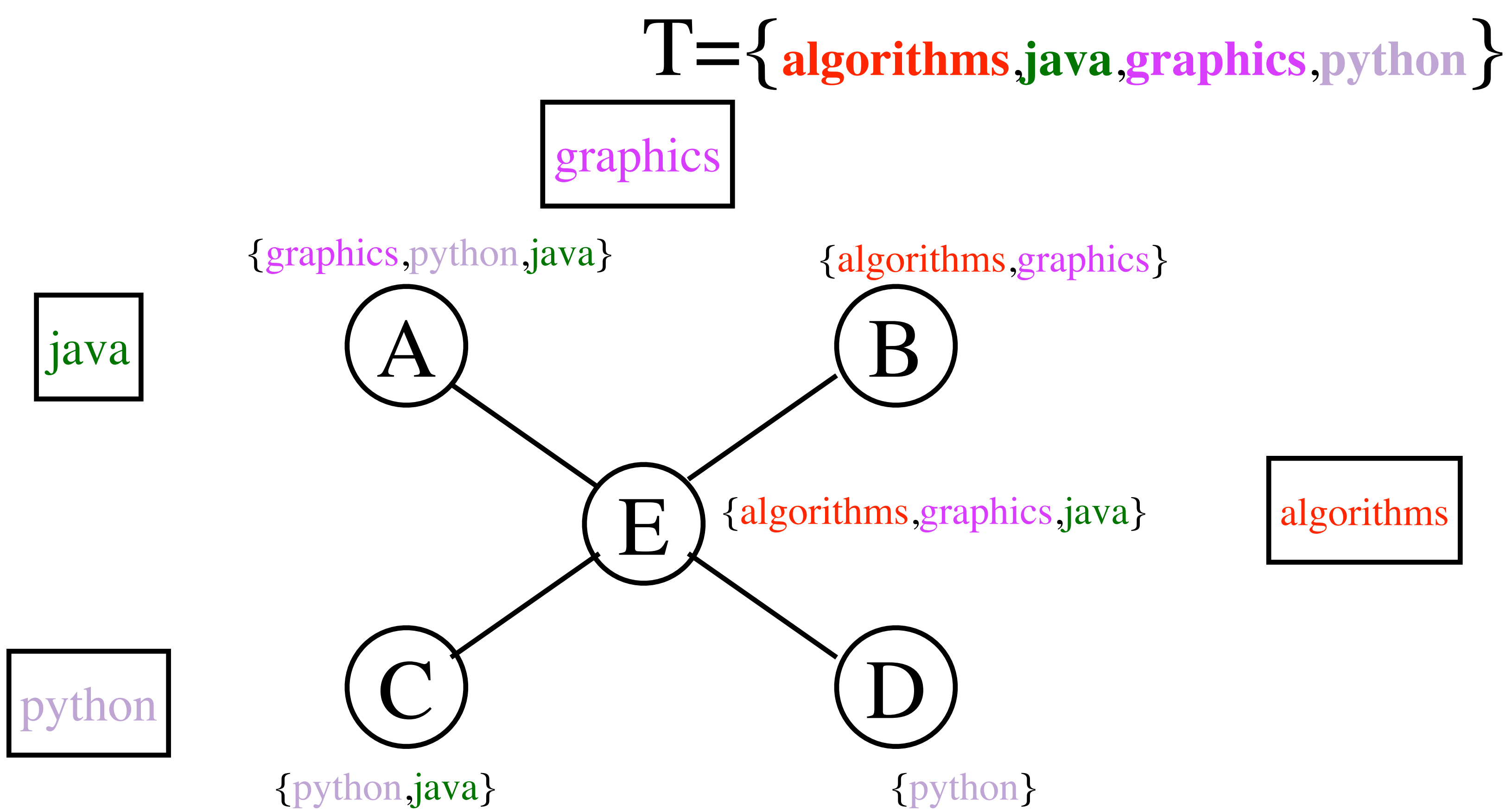
evimaria@cs.bu.edu

The EnhancedSteiner algorithm

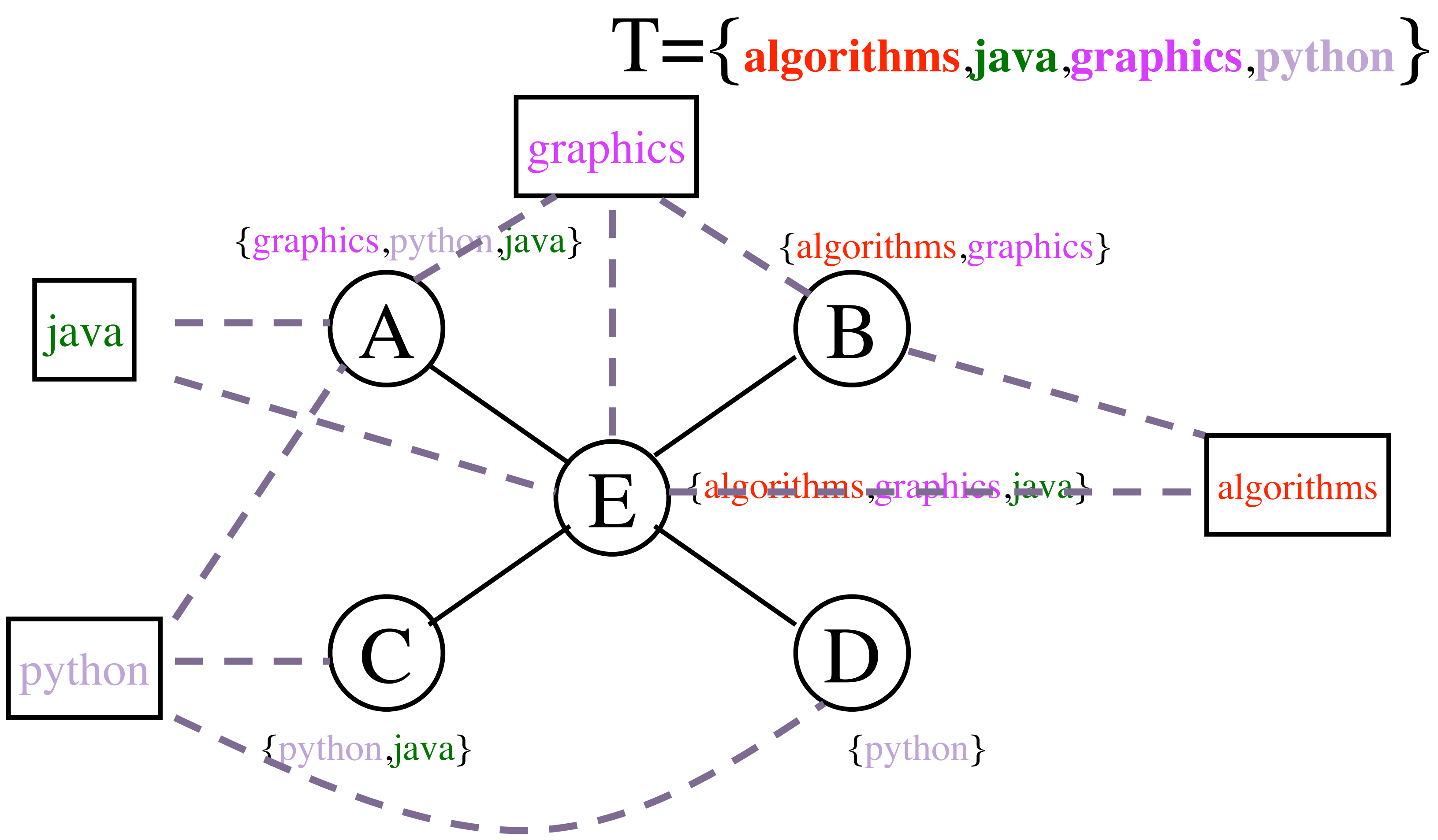
T={algorithms,java,graphics,python}



The EnhancedSteiner algorithm

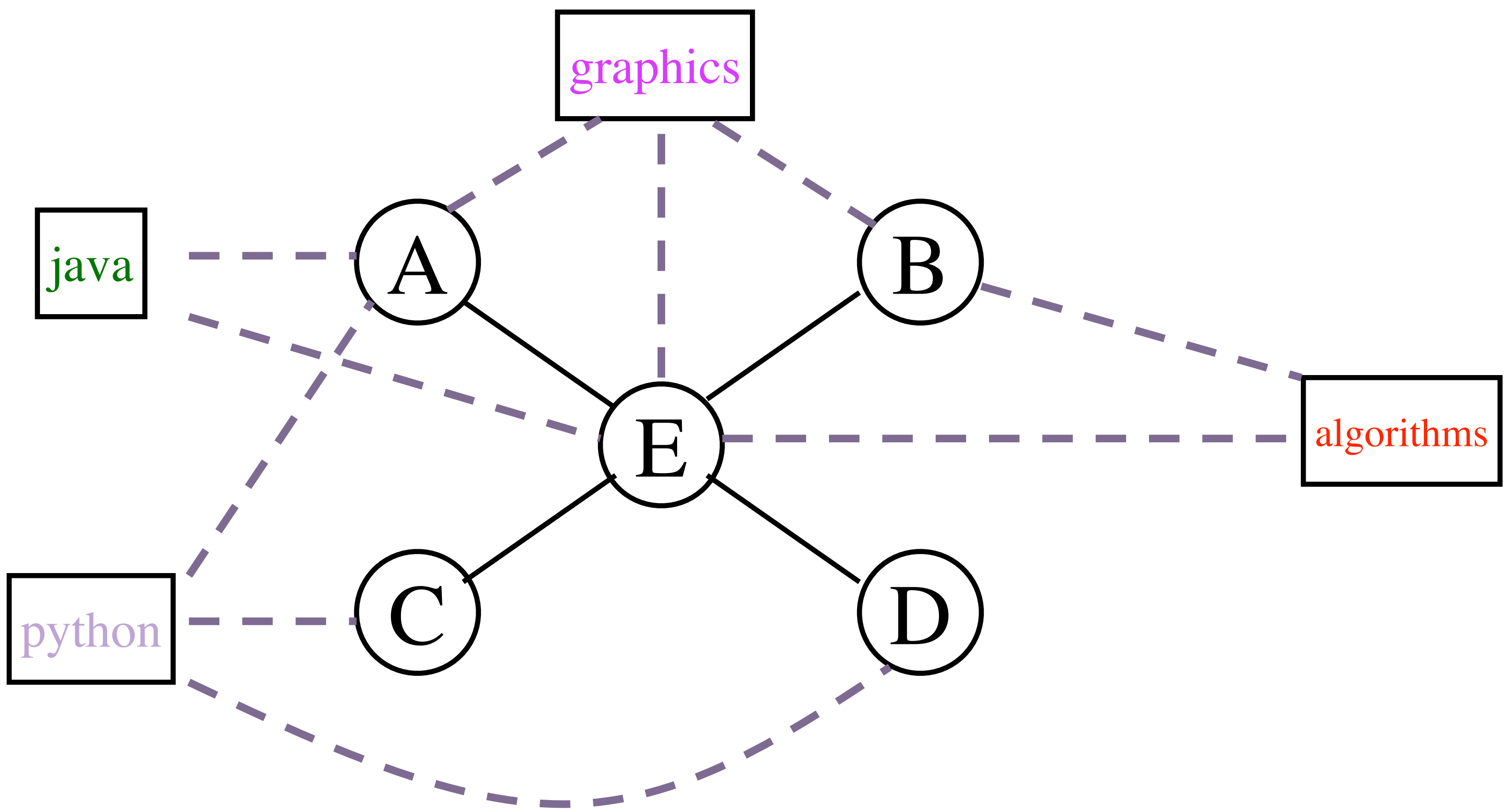


The EnhancedSteiner algorithm

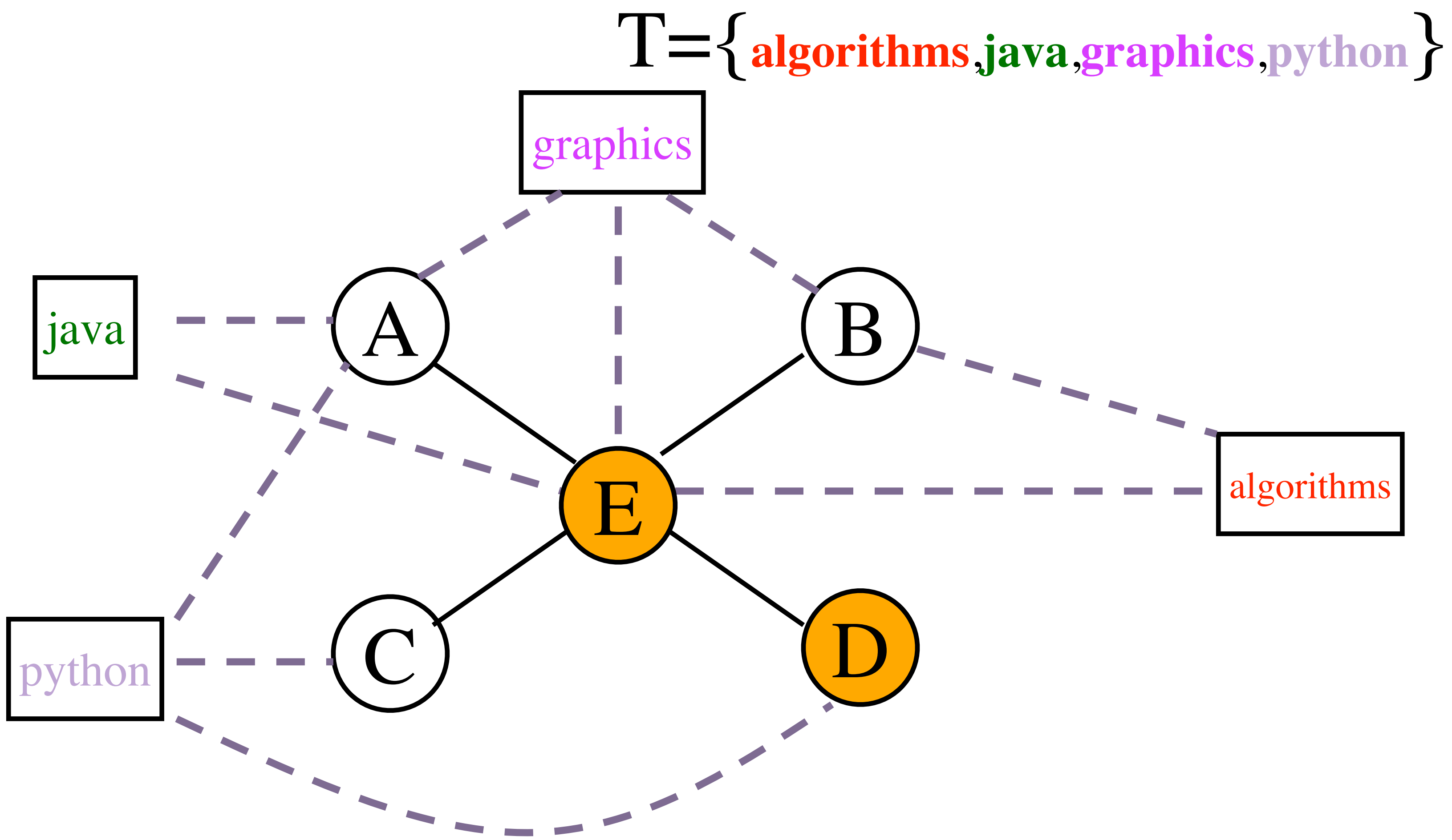


The EnhancedSteiner algorithm

$T = \{$algorithms,java,graphics,python$\}$

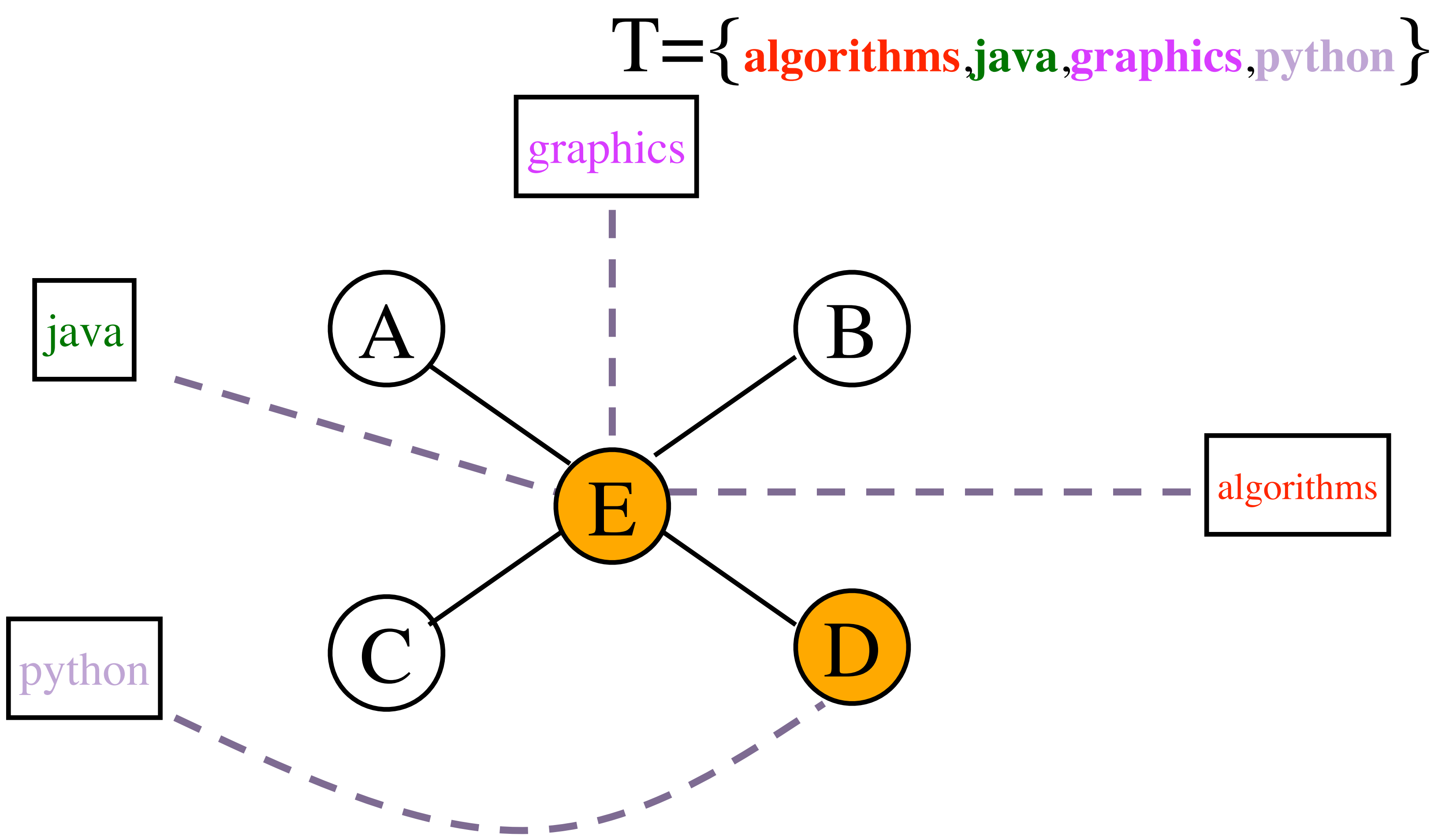


The EnhancedSteiner algorithm

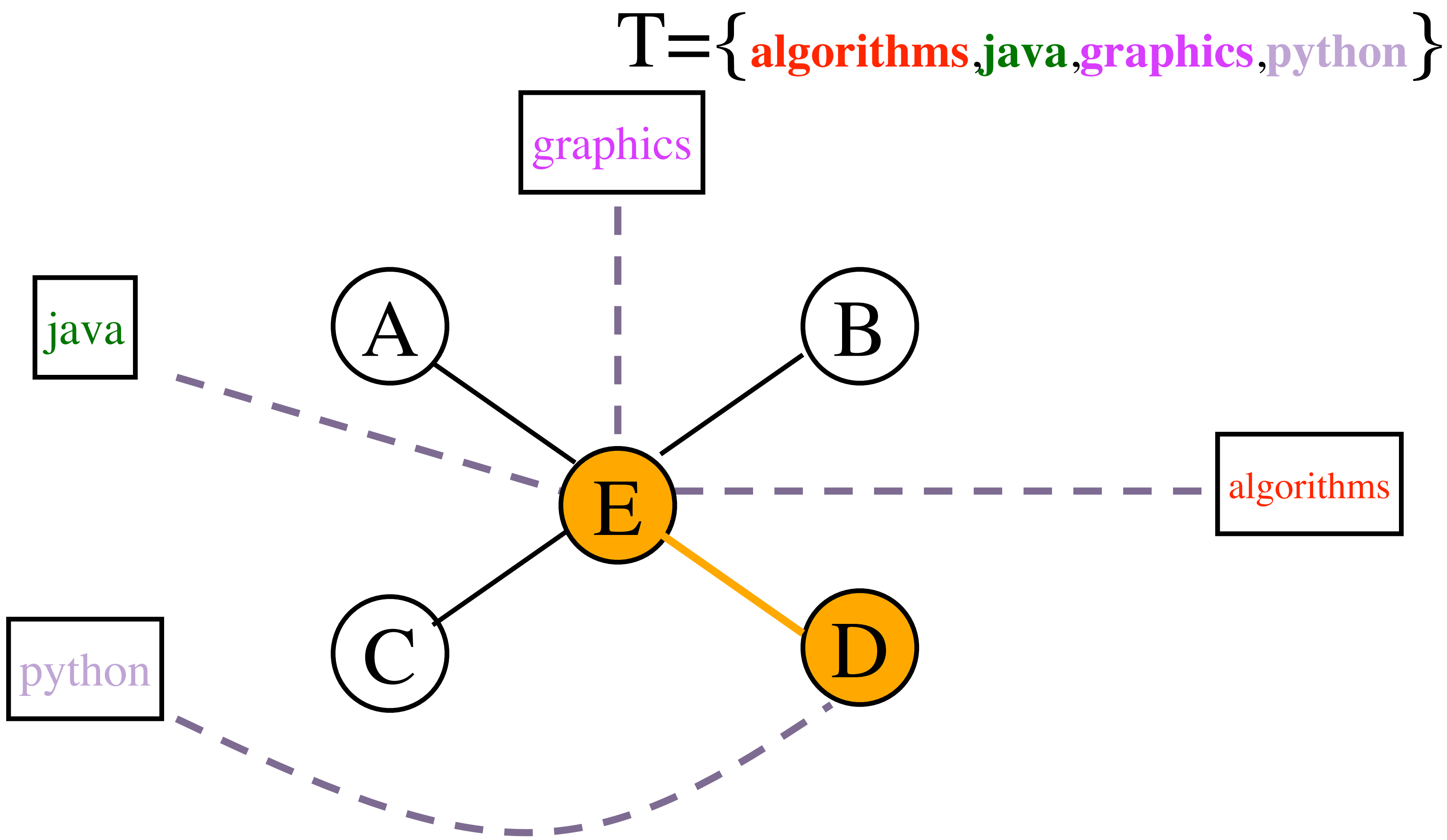


evimaria@cs.bu.edu

The EnhancedSteiner algorithm

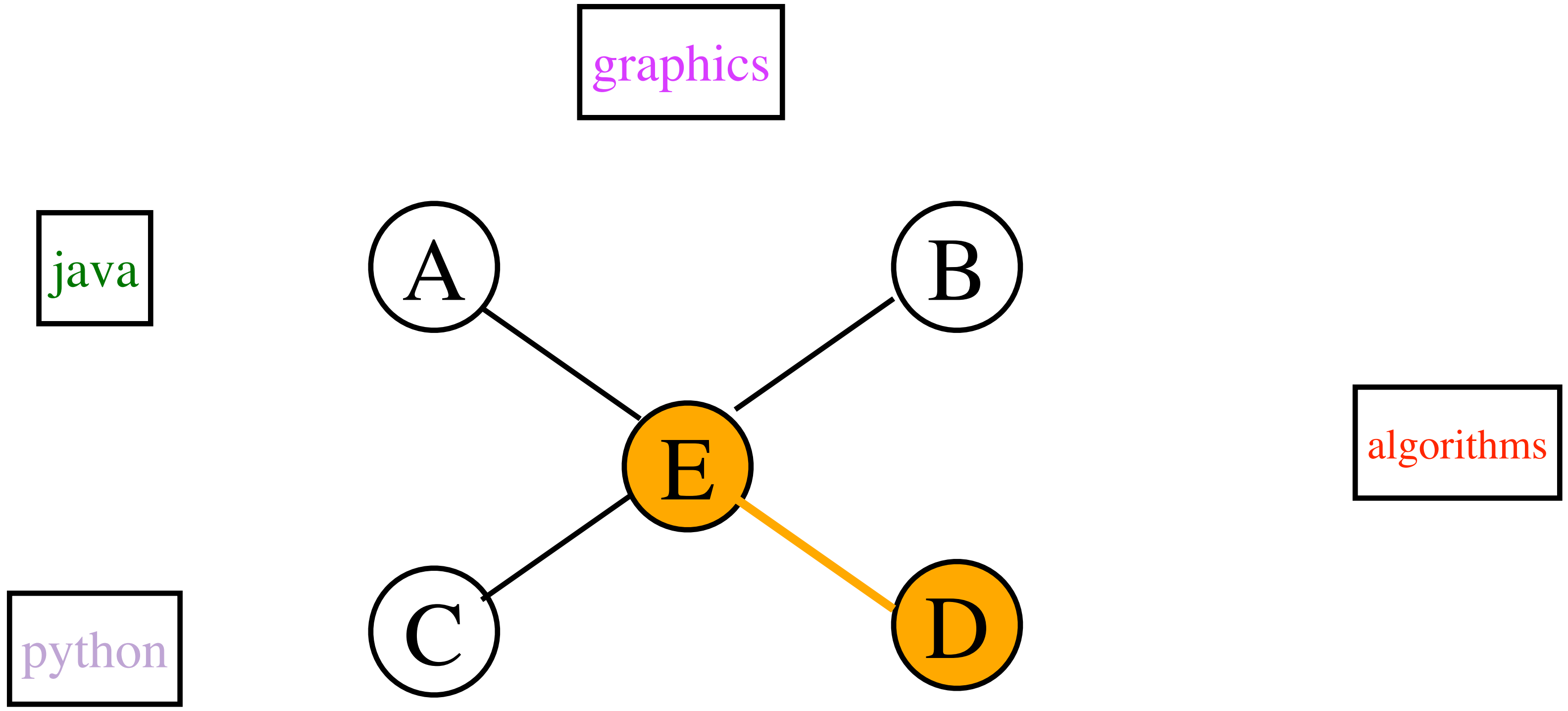


The EnhancedSteiner algorithm



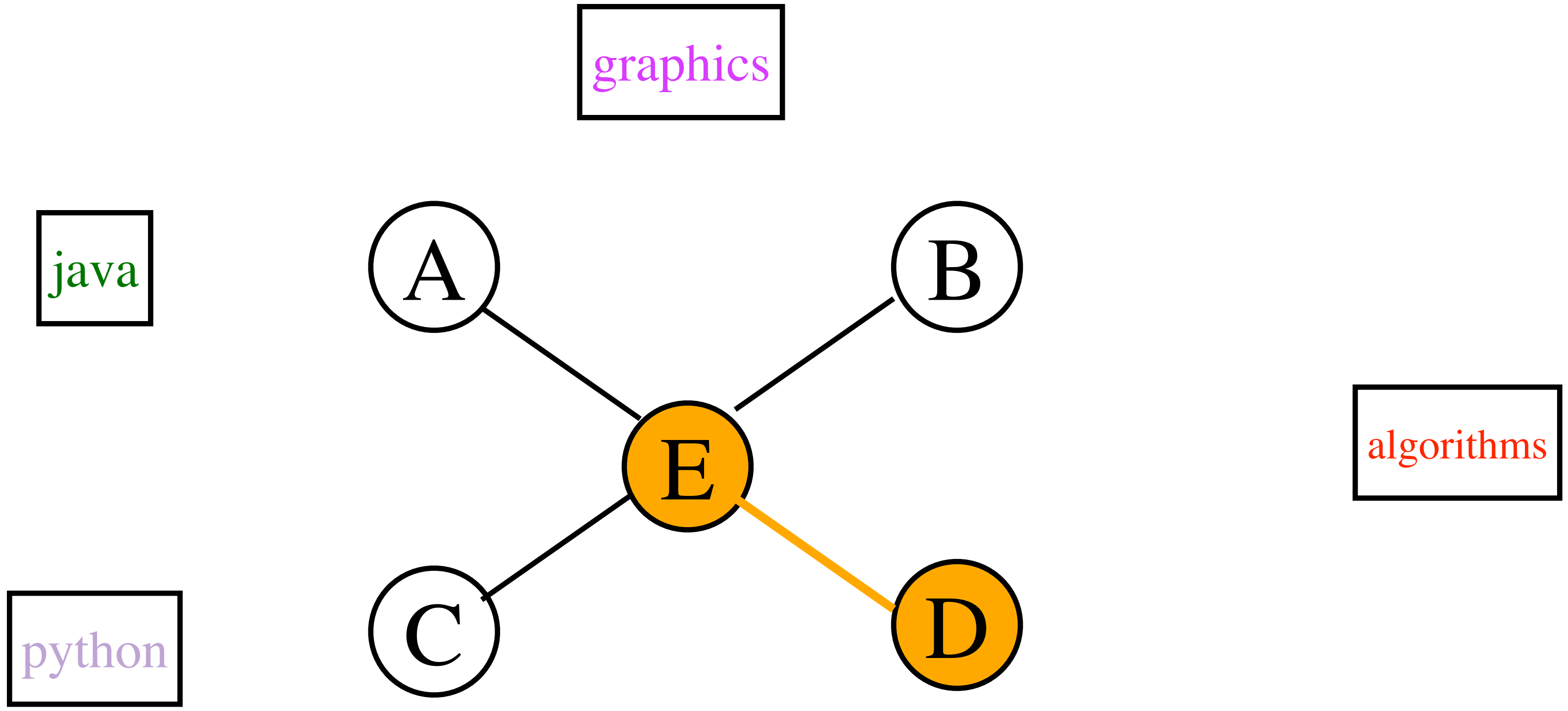
The EnhancedSteiner algorithm

T={algorithms,java,graphics,python}



The EnhancedSteiner algorithm

T={algorithms,java,graphics,python}



MST Cost = 1

Other ways of exploiting the SteinerTree problem

- Graph G(V,E)

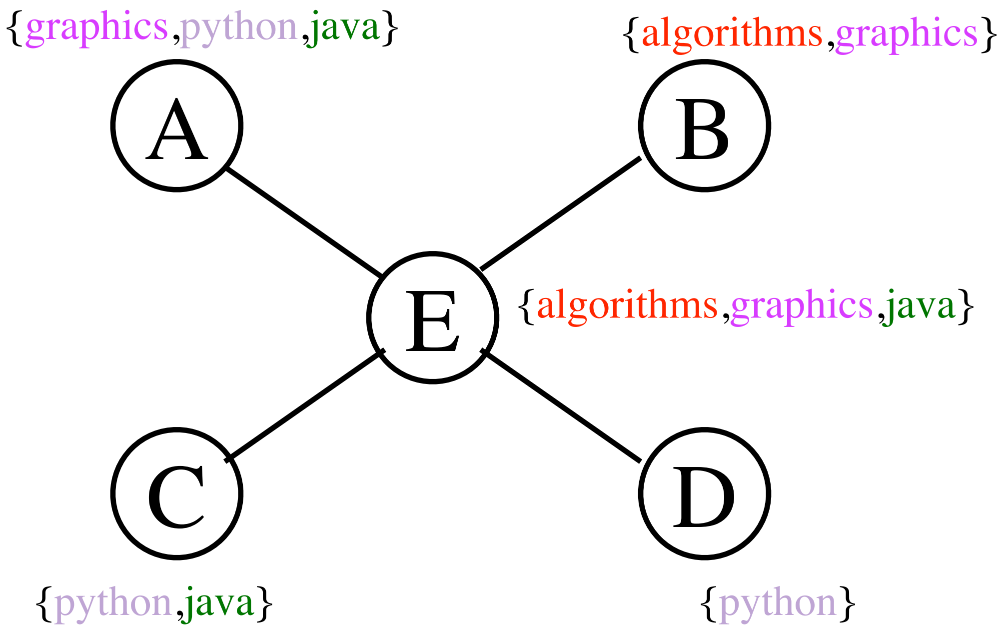
Required vertices

- Partition of V into V = {R,N}

- Find G' subgraph of G such that G' contains all the required vertices (R) and MST(G') is minimized

The CoverSteiner algorithm

T={algorithms,java,graphics,python}

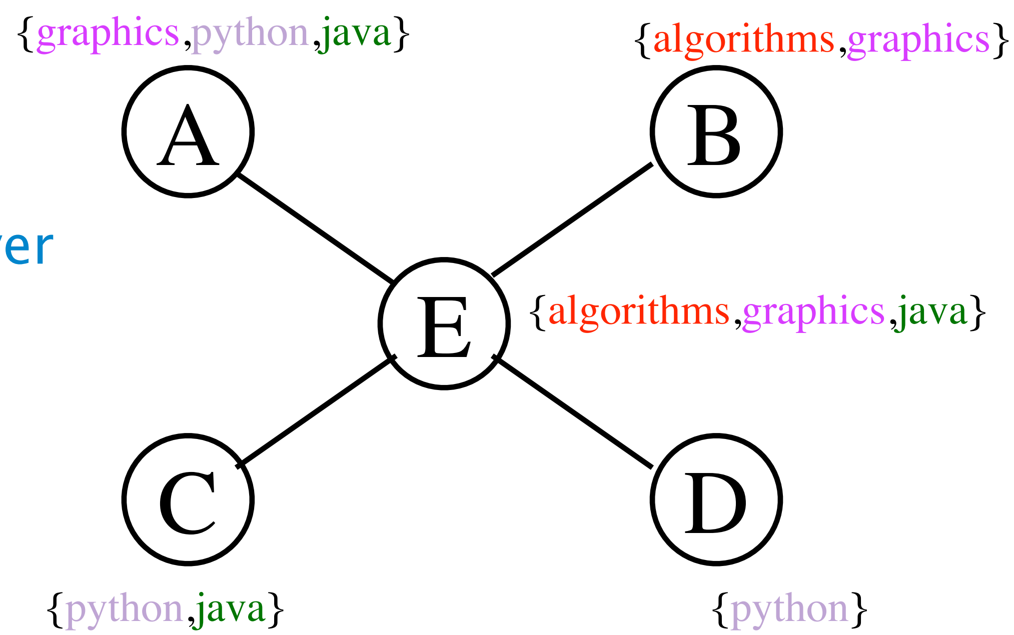


BOSTON UNIVERSITY

The CoverSteiner algorithm

$T=\{$**algorithms**,**java**,**graphics**,**python**$\}$

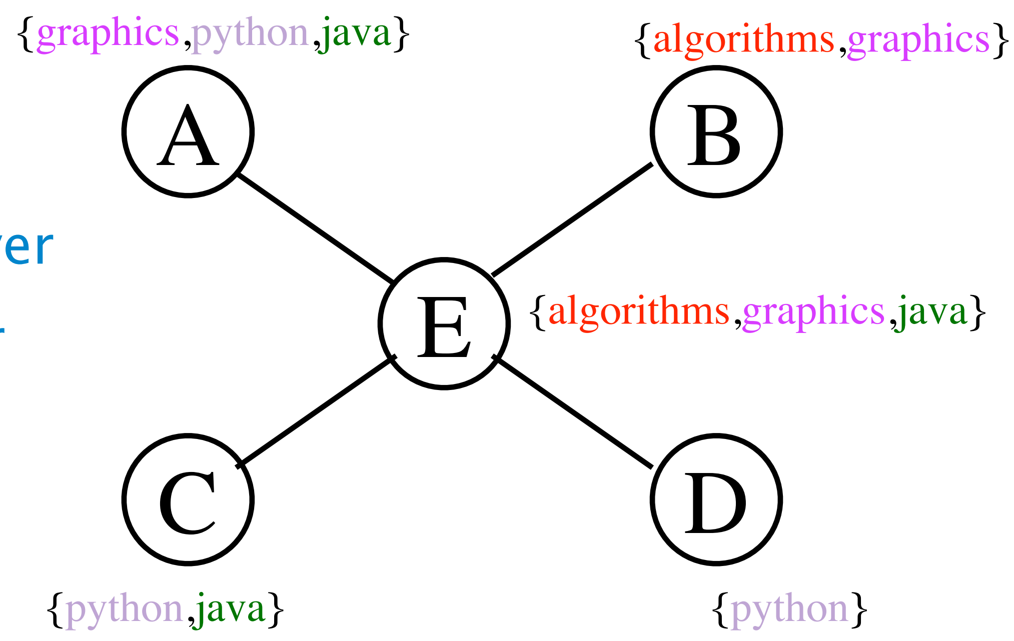
1. Solve SetCover



The CoverSteiner algorithm

T={algorithms,java,graphics,python}

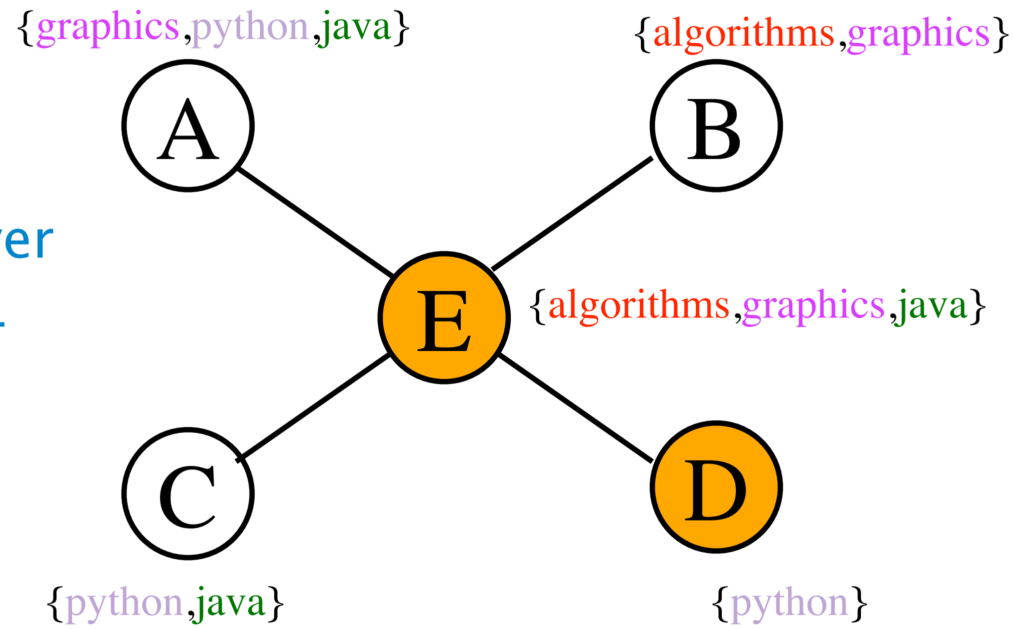
1. Solve SetCover
2. Solve Steiner



The CoverSteiner algorithm

T={algorithms,java,graphics,python}

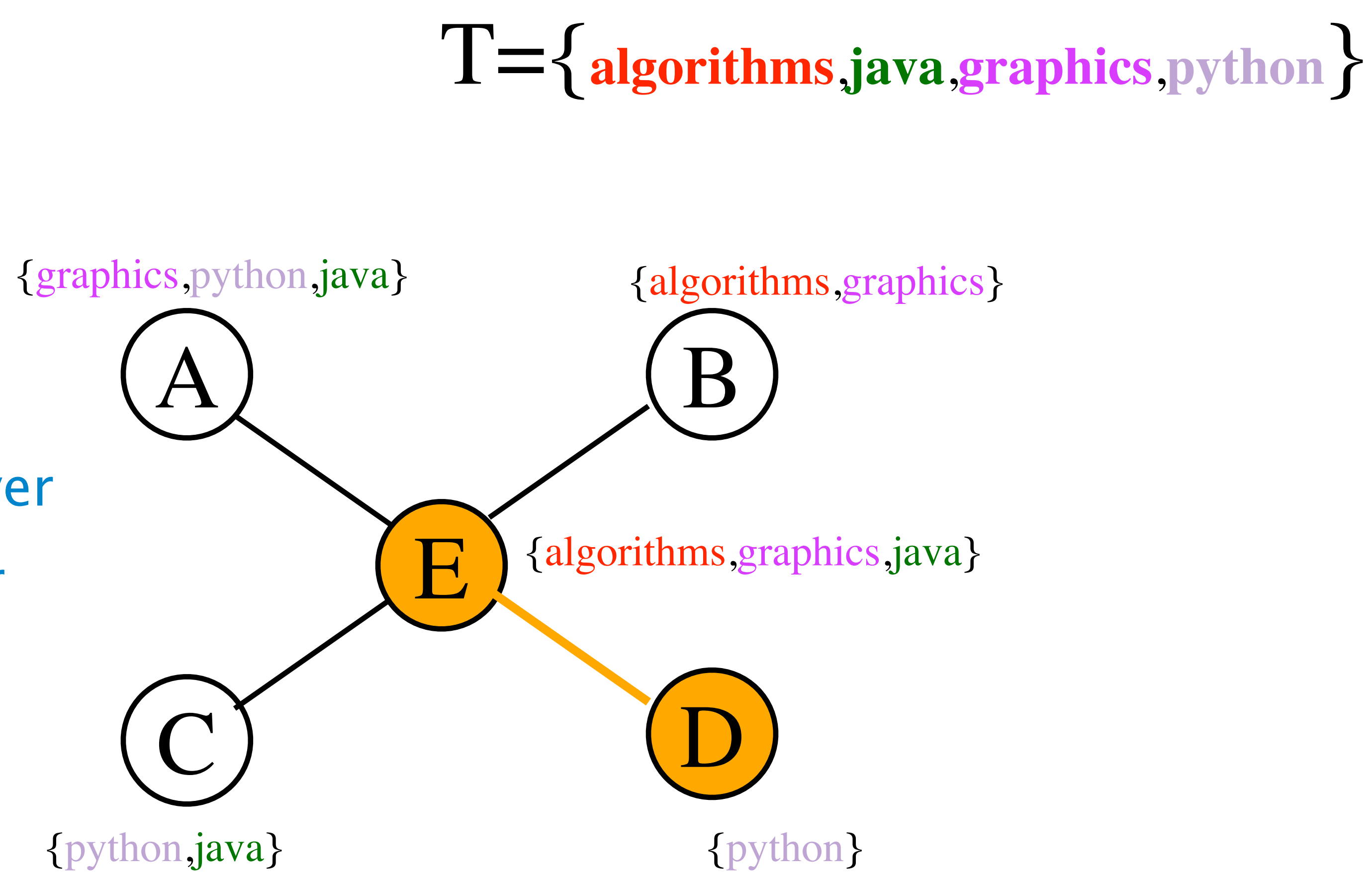
1. Solve SetCover
2. Solve Steiner



evimaria@cs.bu.edu

The CoverSteiner algorithm

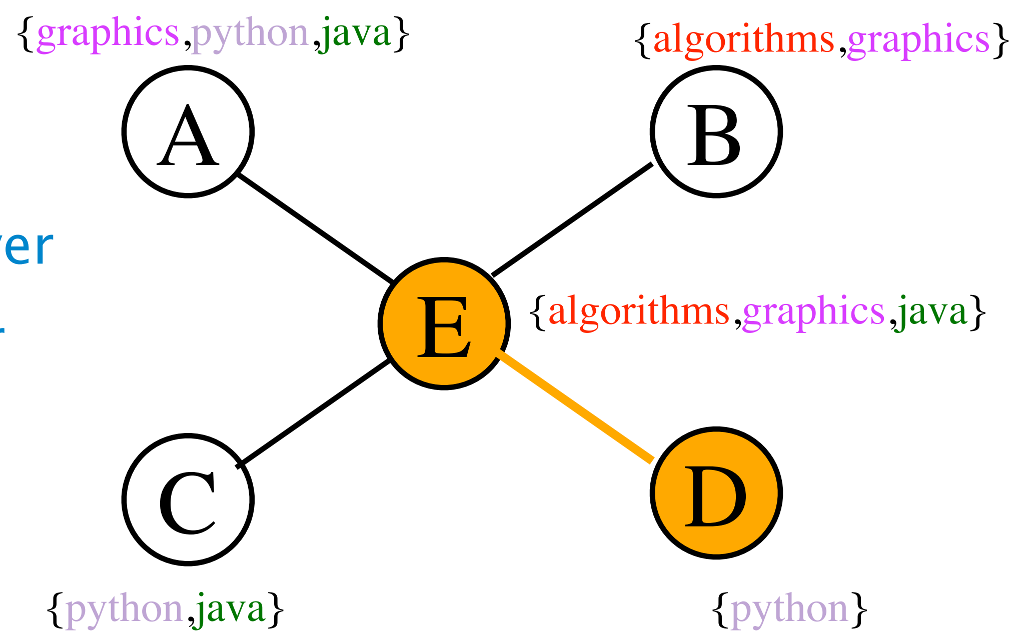
1. Solve SetCover
2. Solve Steiner



The CoverSteiner algorithm

$T = \{algorithms, java, graphics, python\}$

1. Solve SetCover
2. Solve Steiner

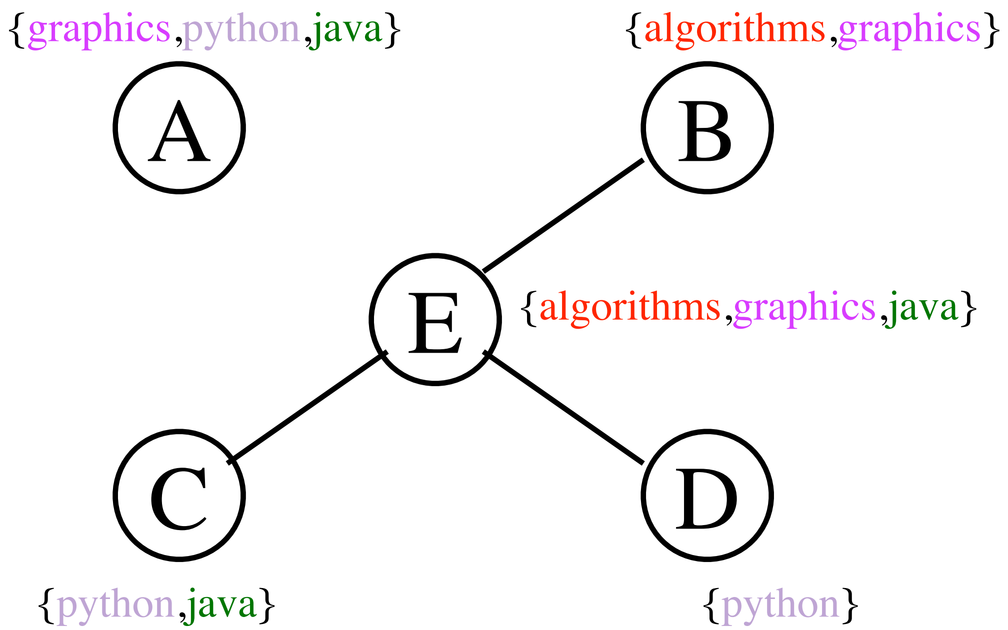


MST Cost = 1

How good is CoverSteiner algorithm?

How good is CoverSteiner algorithm?

T={algorithms,java,graphics,python}

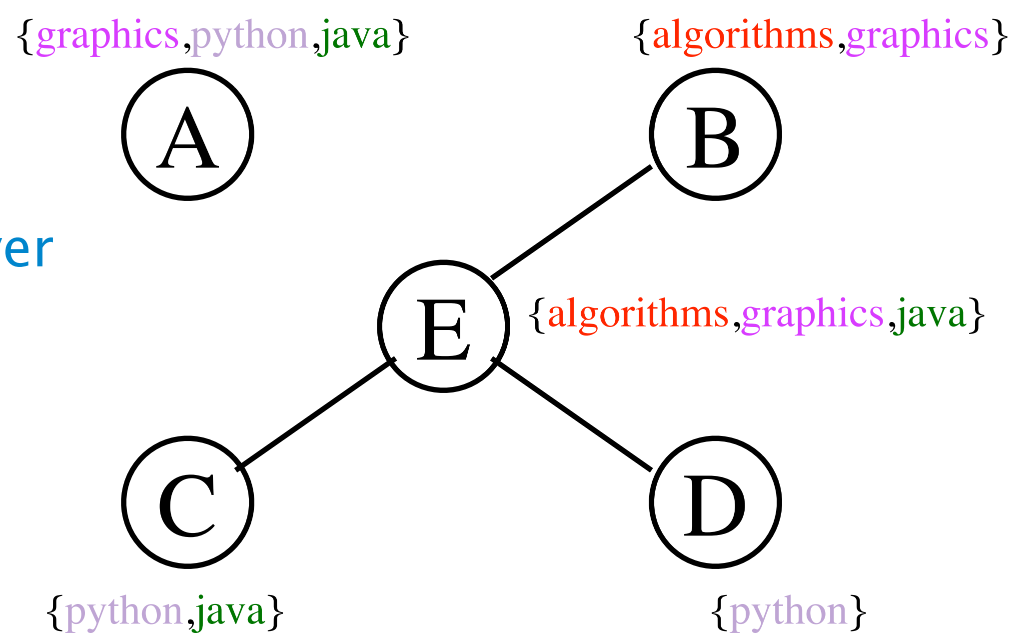


evimaria@cs.bu.edu

How good is CoverSteiner algorithm?

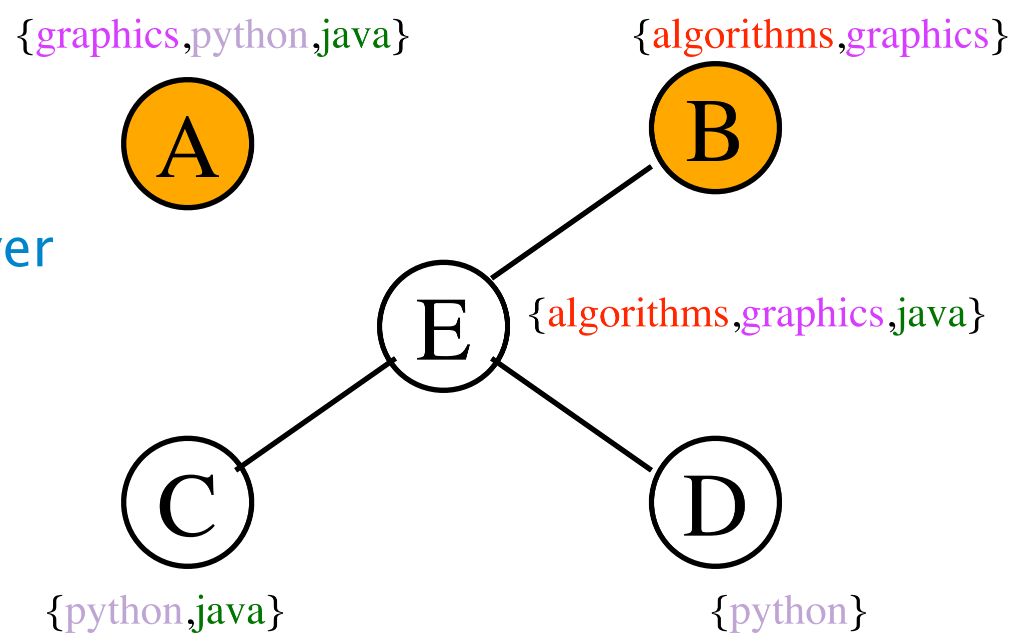
T={algorithms,java,graphics,python}

1. Solve SetCover



How good is CoverSteiner algorithm?

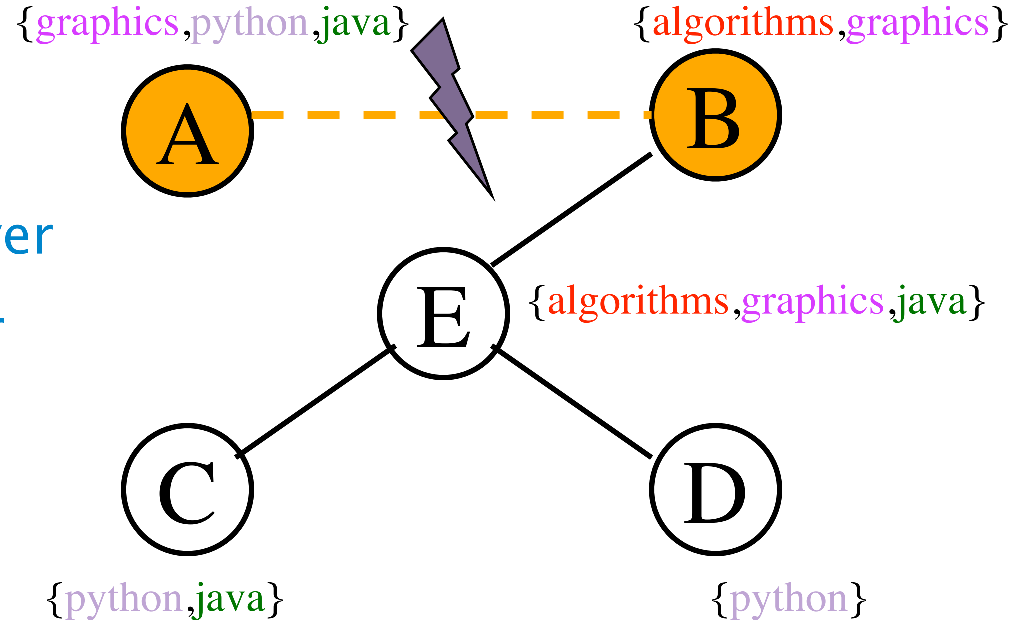
T={algorithms,java,graphics,python}



1. Solve SetCover

How good is CoverSteiner algorithm?

T={algorithms,java,graphics,python}

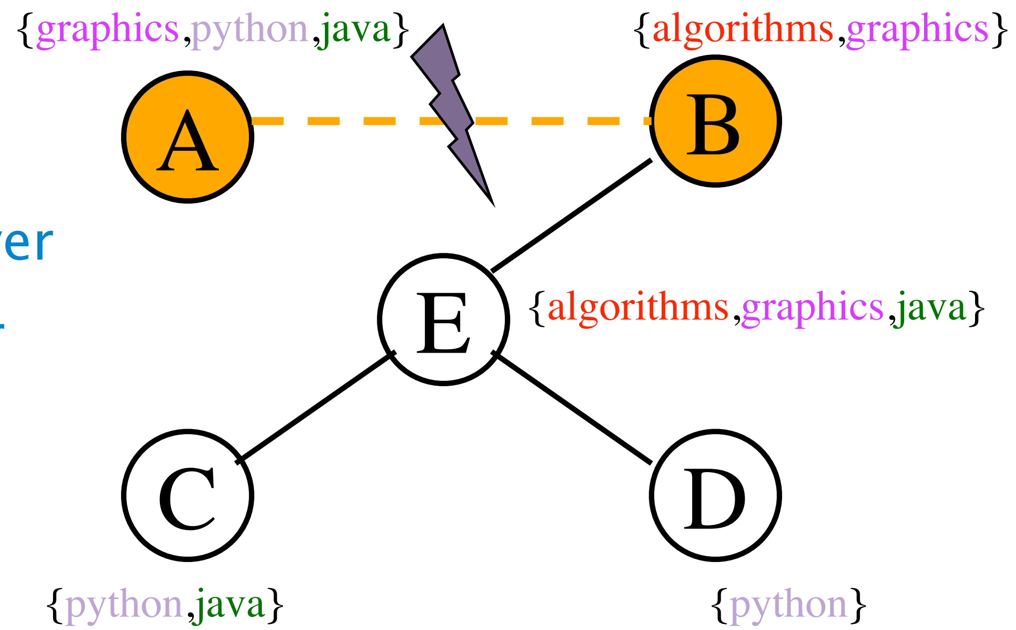


1. Solve SetCover
2. Solve Steiner

How good is CoverSteiner algorithm?

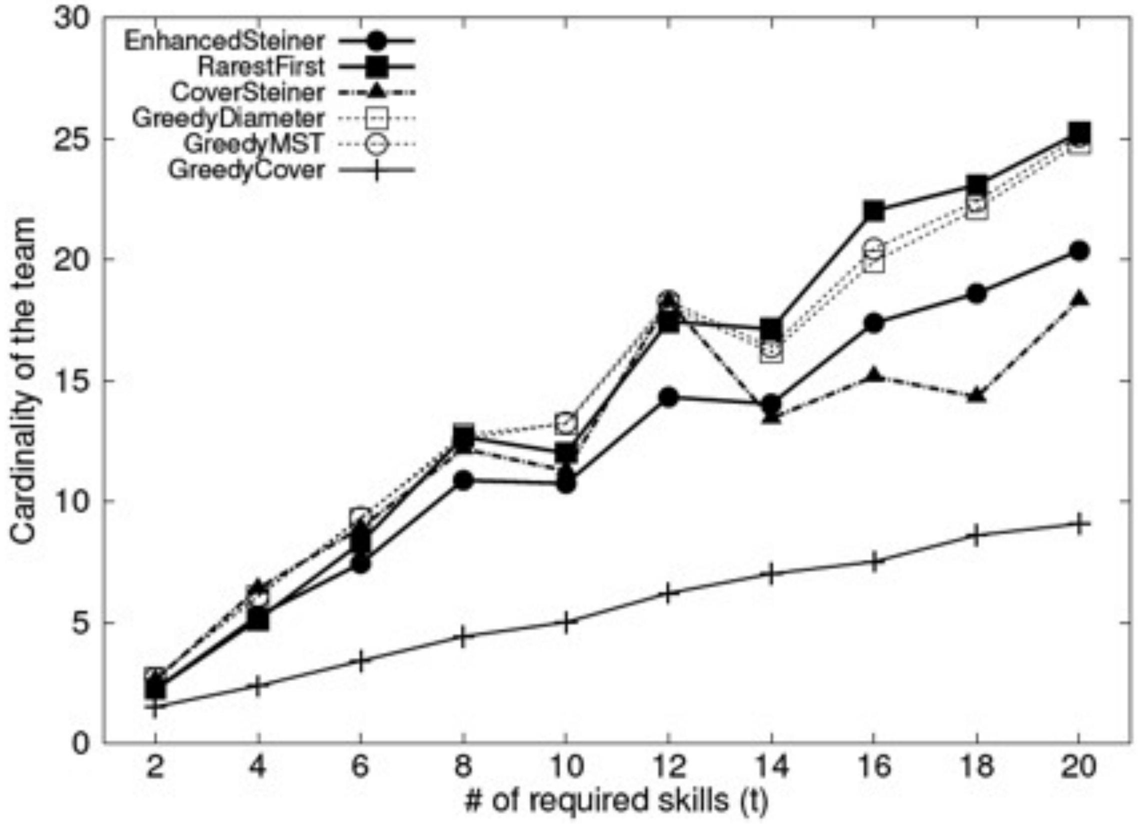
T={algorithms,java,graphics,python}

1. Solve SetCover
2. Solve Steiner



MST Cost = Infty

Experiments – Cardinality of teams



Dataset

DBLP graph (DB, Theory, ML, DM)

~6000 authors

~2000 features

Features: keywords appearing in papers

Tasks: Subsets of keywords with different cardinality **k**

BOSTON UNIVERSITY

Example teams (I)

- S. Brin, L. Page: The anatomy of a large-scale hypertextual Web search engine
  - **Paolo Ferragina, Patrick Valduriez, H. V. Jagadish, Alon Y. Levy, Daniela Florescu** Divesh Srivastava, S. Muthukrishnan
  - **P. Ferragina ,J. Han, H. V.Jagadish, Kevin Chen-Chuan Chang, A. Gulli**, S. Muthukrishnan, Laks V. S. Lakshmanan

evimaria@cs.bu.edu

Example teams (II)

- J. Han, J. Pei, Y. Yin: Mining frequent patterns without candidate generation
  - F. Bronchi
  - A. Gionis, H. Mannila, R. Motwani

evimaria@cs.bu.edu

Extensions

- Skill attribution

	Team	Skill Attribution
Experts' skills	Known	Unknown
Participation of experts in	Unknown	Known
Network structure	Known	Irrelevant

- Team chemistry as a factor of success

evimaria@cs.bu.edu

Example teams (II)

- J. Han, J. Pei, Y. Yin: Mining frequent patterns without candidate generation
  - F. Bronchi
  - A. Gionis, H. Mannila, R. Motwani

evimaria@cs.bu.edu