Fourth ACM International Conference on Web Search and Data Mining
9-12 February 2011
Hong Kong

Claudio Lucchese *Salvatore Orlando* *Raffaele Perego* *Fabrizio Silvestri* *Gabriele Tolomei*

Identifying Task-based Sessions in Search Engine Query Logs

Ca' Foscari University of Venice
ISTI-CNR, Pisa
Italy

February, 12 2011

Agenda

- Introduction
- Contributions
- Experiments and Results
- Conclusions and Future Work

Agenda

- Introduction
- Contributions
- Experiments and Results
- Conclusions and Future Work

Problem Statement: TSDP

Task-based Session Discovery Problem:

Discover sets of possibly non contiguous queries issued by users and collected by Web Search Engine Query Logs whose aim is to carry out specific "tasks"

Background

- What is a Web task?

Background

- What is a Web task?
  - A "template" for representing any (atomic) activity that can be achieved by exploiting the information available on the Web, e.g., "find a recipe", "book a flight", "read news", etc.

Background

- What is a Web task?
  - A "template" for representing any (atomic) activity that can be achieved by exploiting the information available on the Web, e.g., "find a recipe","book a flight","read news", etc.
- Why WSE Query Logs?

Background

- What is a Web task?
  - A "template" for representing any (atomic) activity that can be achieved by exploiting the information available on the Web, e.g., "find a recipe","book a flight","read news", etc.
- Why WSE Query Logs?
  - Users rely on WSEs for satisfying their information needs by issuing possibly interleaved stream of related queries

Background

- What is a Web task?
  - A "template" for representing any (atomic) activity that can be achieved by exploiting the information available on the Web, e.g., "find a recipe","book a flight","read news", etc.
- Why WSE Query Logs?
  - Users rely on WSEs for satisfying their information needs by issuing possibly interleaved stream of related queries
  - WSEs collect the search activities, i.e., sessions, of their users by means of issued queries, timestamps, clicked results, etc.

Background

- What is a Web task?
  - A "template" for representing any (atomic) activity that can be achieved by exploiting the information available on the Web, e.g., "find a recipe", "book a flight", "read news", etc.
- Why WSE Query Logs?
  - Users rely on WSEs for satisfying their information needs by issuing possibly interleaved stream of related queries
  - WSEs collect the search activities, i.e., sessions, of their users by means of issued queries, timestamps, clicked results, etc.
  - User search sessions (especially long-term ones) might contain interesting patterns that can be mined, e.g., sub-sessions whose queries aim to perform the same Web task

Motivation

- "Addiction to Web search": no matter what your information need is, ask it to a WSE and it will give you the answer, e.g., people querying Google for "google"!

Motivation

- "Addiction to Web search": no matter what your information need is, ask it to a WSE and it will give you the answer, e.g., people querying Google for "google"!
- Everyone who is now at WSDM 2011 has dealt with a lot of "stuff" for organizing her/his attendance

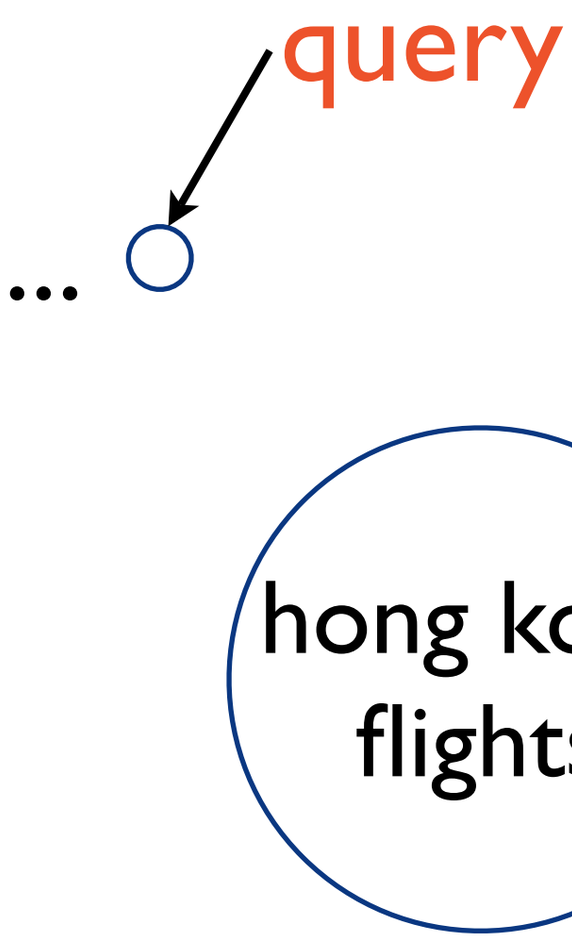
Motivation

- "Addiction to Web search": no matter what your information need is, ask it to a WSE and it will give you the answer, e.g., people querying Google for "google"!
- Everyone who is now at WSDM 2011 has dealt with a lot of "stuff" for organizing her/his attendance
- Conference Web site is full of useful information but still some tasks have to be performed (e.g., book flight, reserve hotel room, rent car, etc.)

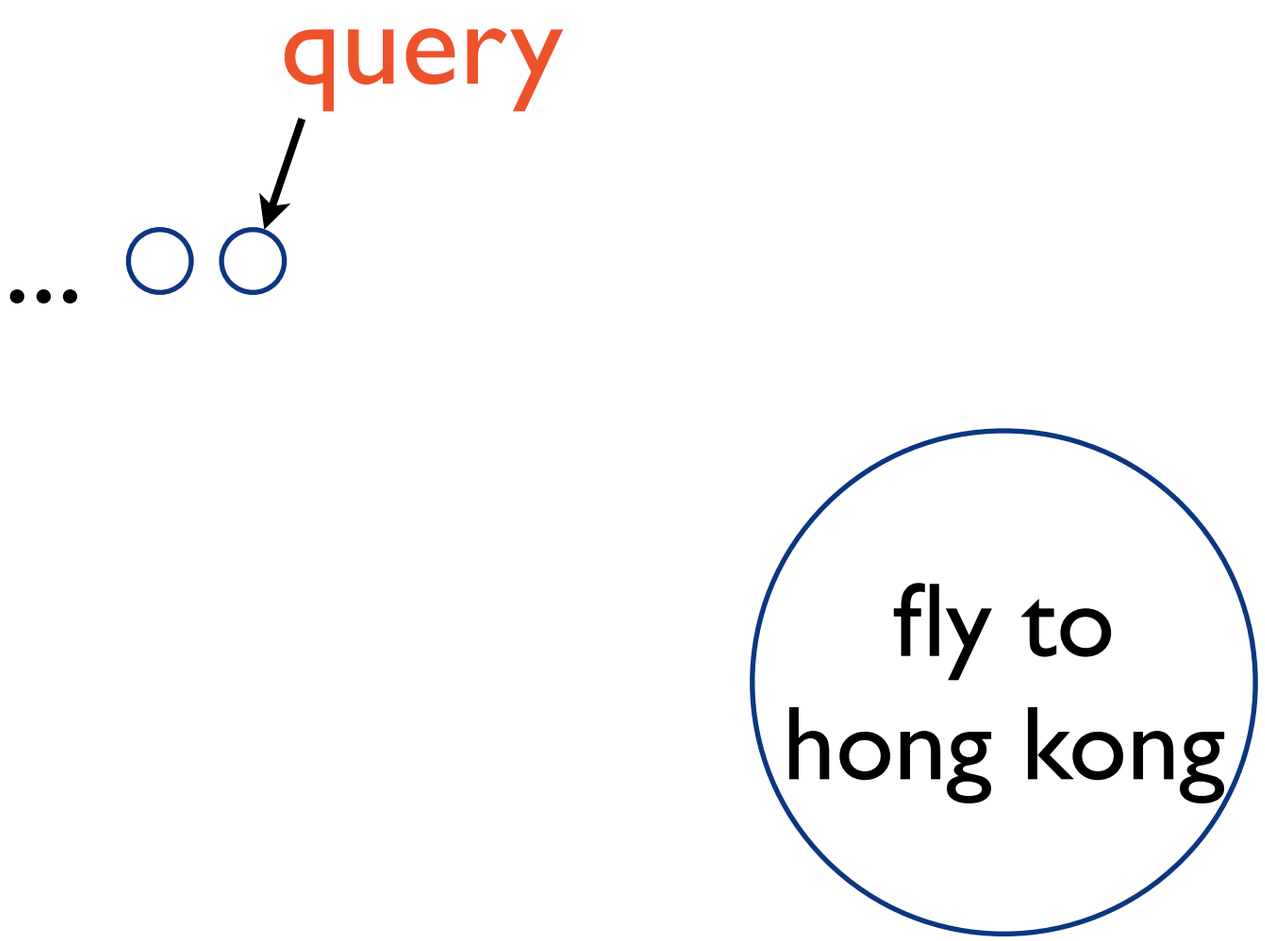
Motivation

- "Addiction to Web search": no matter what your information need is, ask it to a WSE and it will give you the answer, e.g., people querying Google for "google"!
- Everyone who is now at WSDM 2011 has dealt with a lot of "stuff" for organizing her/his attendance
- Conference Web site is full of useful information but still some tasks have to be performed (e.g., book flight, reserve hotel room, rent car, etc.)
- Discovering tasks from WSE logs will allow us to better understand user search intents at a "higher level of abstraction":
  - from query-by-query to task-by-task Web search

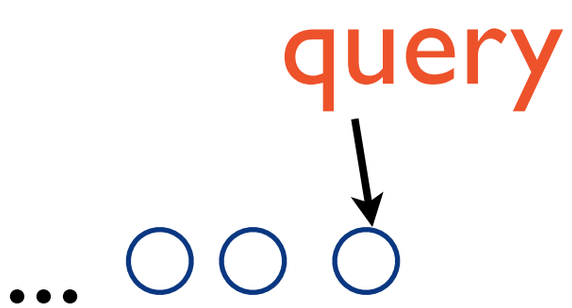
The Big Picture



The Big Picture



The Big Picture

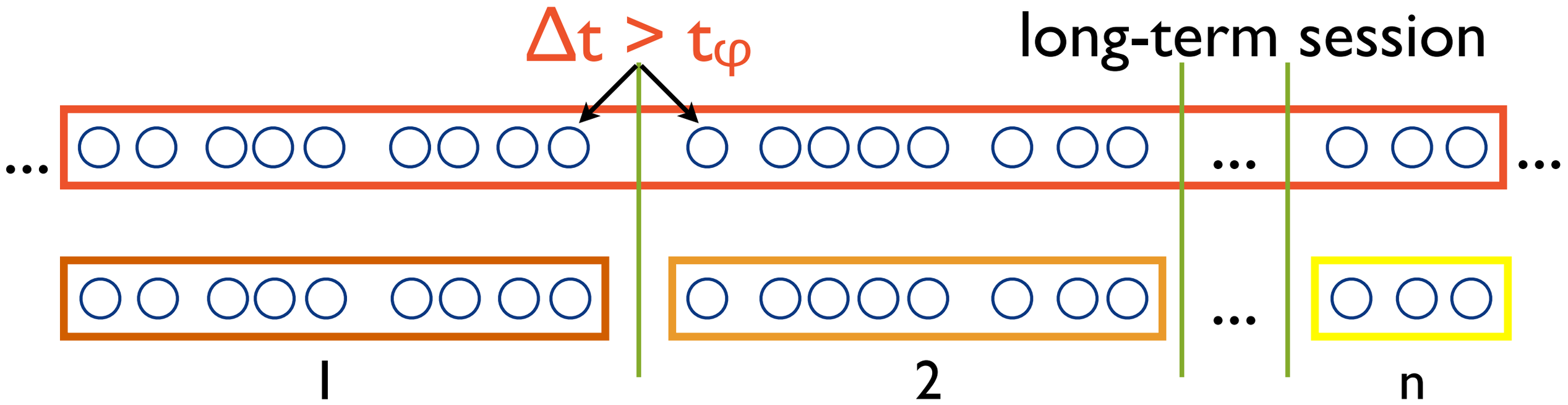


The Big Picture

The Big Picture

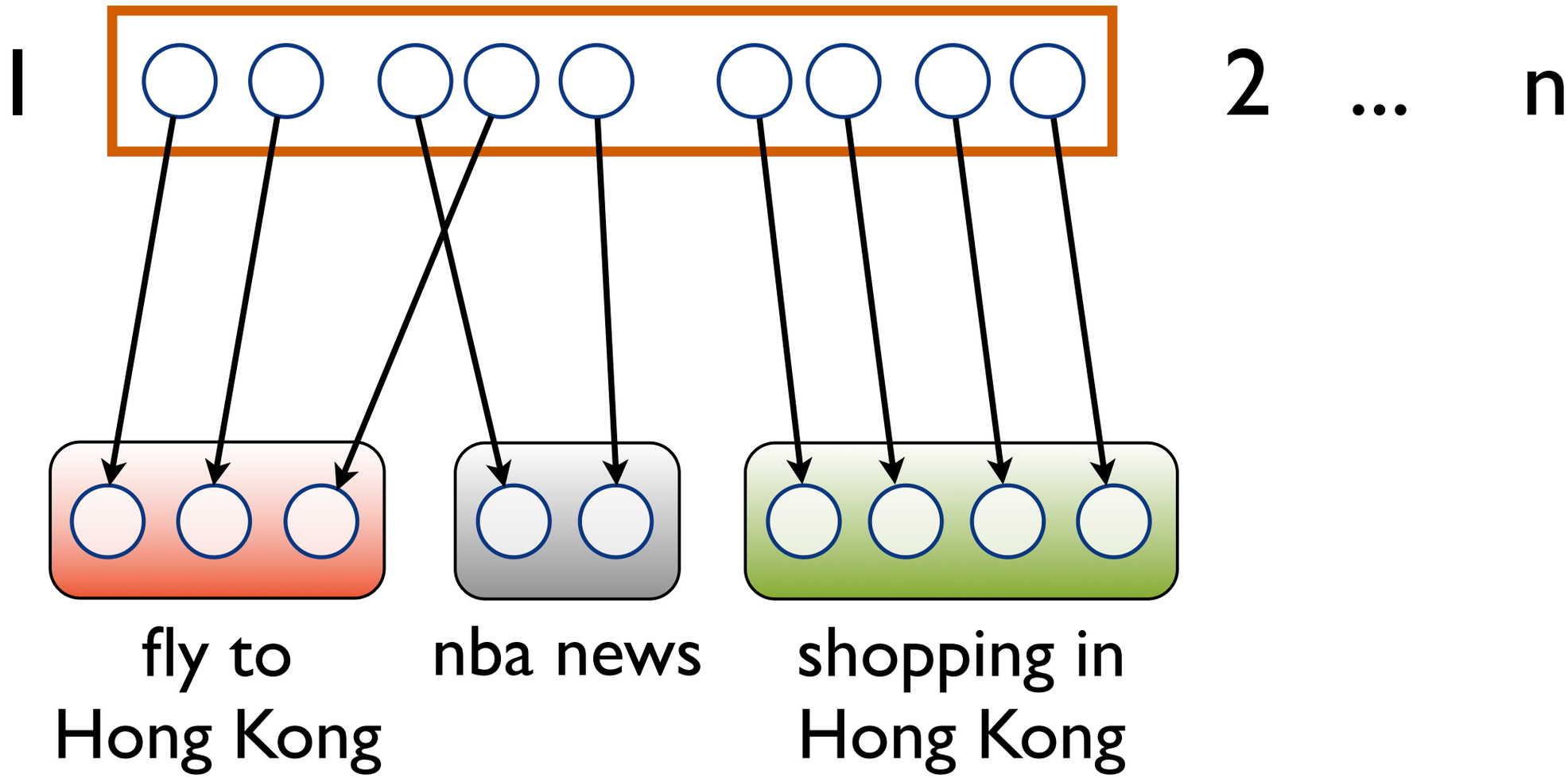
long-term session

The Big Picture



The Big Picture

The Big Picture



Related Work

- Previous work on session identification can be classified into:

1. time-based
2. content-based
3. novel heuristics (combining 1. and 2.)

Related Work: time-based

- 1999: Silverstein *et al.* [1] firstly defined the concept of "session":
  - 2 adjacent queries (q_i, q_{i+1}) are part of the same session if their time submission gap is at most 5 minutes
- 2000: He and Göker [2] used different timeouts to split user sessions (from 1 to 50 minutes)
- 2006: Jansen and Spink [4] described a session as the time gap between the first and last recorded timestamp on the WSE server

Related Work: time-based

- 1999: Silverstein *et al.* [1] firstly defined the concept of "session":
  - 2 adjacent queries (q_i, q_{i+1}) are part of the same session if their time submission gap is at most 5 minutes
- 2000: He and Göker [2] used different timeouts to split user sessions (from 1 to 50 minutes)
- 2006: Jansen and Spink [4] described a session as the time gap between the first and last recorded timestamp on the WSE server

PROs

✓ ease of implementation

CONs

✓ unable to deal with multi-tasking behaviors

Related Work: content-based

- Some work exploit lexical content of the queries for determining a topic shift in the stream, i.e., session boundary [3, 5, 6, 7]

- Several string similarity scores have been proposed, e.g., Levenshtein, Jaccard, etc.

- 2005: Shen *et al.* [8] compared "expanded representation" of queries

  - expansion of a query q is obtained by concatenating titles and Web snippets for the top-50 results provided by a WSE for q

Related Work: content-based

- Some work exploit lexical content of the queries for determining a topic shift in the stream, i.e., session boundary [3, 5, 6, 7]

- Several string similarity scores have been proposed, e.g., Levenshtein, Jaccard, etc.

- 2005: Shen *et al.* [8] compared "expanded representation" of queries

  - expansion of a query q is obtained by concatenating titles and Web snippets for the top-50 results provided by a WSE for q

PROs

✓ effectiveness improvement

CONs

✓ *vocabulary-mismatch problem*: e.g., ("nba","kobe bryant")

Related Work: novel

- 2005: Radlinski and Joachims [3] introduced query chains, i.e., sequence of queries with similar information need

- 2008: Boldi *et al.* [9] introduce the query-flow graph as a model for representing WSE log data
  - session identification as Traveling Salesman Problem

- 2008: Jones and Klinkner [10] address a problem similar to the TSDP
  - hierarchical search: mission vs. goal
  - supervised approach: learn a suitable binary classifier to detect whether two queries (q_i, q_j) belong to the same task or not

Related Work: novel

- 2005: Radlinski and Joachims [3] introduced query chains, i.e., sequence of queries with similar information need

- 2008: Boldi *et al.* [9] introduce the query-flow graph as a model for representing WSE log data
  - session identification as Traveling Salesman Problem

- 2008: Jones and Klinkner [10] address a problem similar to the TSDP
  - hierarchical search: mission vs. goal
  - supervised approach: learn a suitable binary classifier to detect whether two queries (q_i, q_j) belong to the same task or not

PROs

✓ effectiveness improvement

CONs

✓ computational complexity

Agenda

- Introduction
- Contributions
- Experiments and Results
- Conclusions and Future Work

Outline

- Formalize the Task-based Session Discovery Problem

Outline

- Formalize the Task-based Session Discovery Problem
- Analyze a real long-term WSE log of queries

Outline

- Formalize the Task-based Session Discovery Problem
- Analyze a real long-term WSE log of queries
- Build a ground-truth of tasks by manually grouping a sample of task-related queries in the given WSE log

Outline

- Formalize the Task-based Session Discovery Problem
- Analyze a real long-term WSE log of queries
- Build a ground-truth of tasks by manually grouping a sample of task-related queries in the given WSE log
- Perform some statistics on top of the ground-truth

Outline

- Formalize the Task-based Session Discovery Problem
- Analyze a real long-term WSE log of queries
- Build a ground-truth of tasks by manually grouping a sample of task-related queries in the given WSE log
- Perform some statistics on top of the ground-truth
- Propose several techniques for addressing the TSDP

Agenda

- Introduction
- Contributions
  - Query Log Analysis
  - Ground-truth Analysis
  - Approaching TSDP
- Experiments and Results
- Conclusions and Future Work

Data Set: AOL Query Log

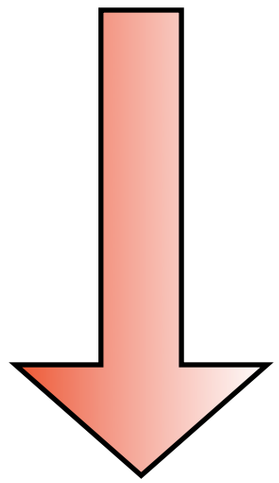
Original Data Set

- ✓ 3-months collection
- ✓ ~20M queries
- ✓ ~657K users

Data Set: AOL Query Log

Original Data Set

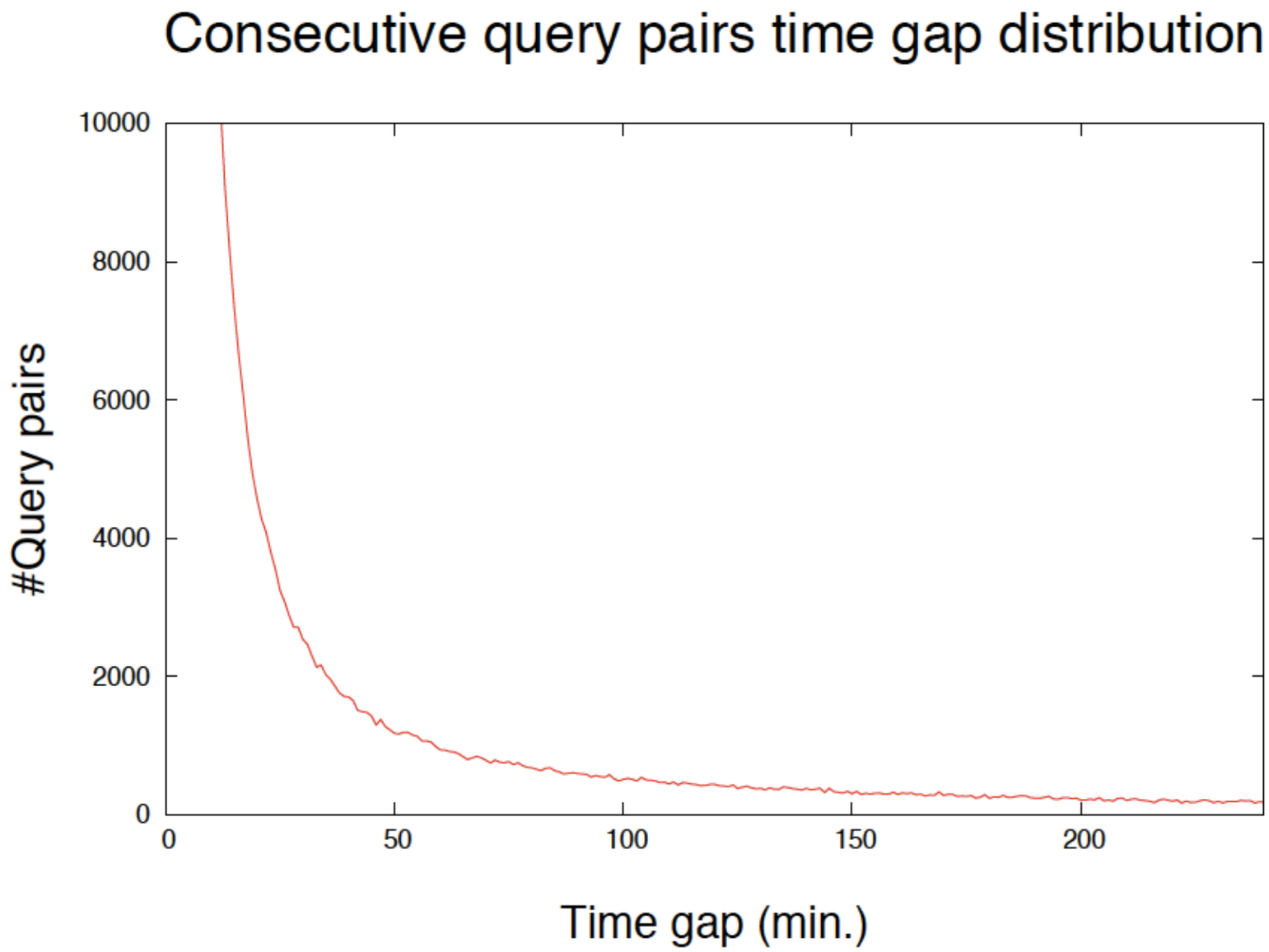
- ✓ 3-months collection
- ✓ ~20M queries
- ✓ ~657K users



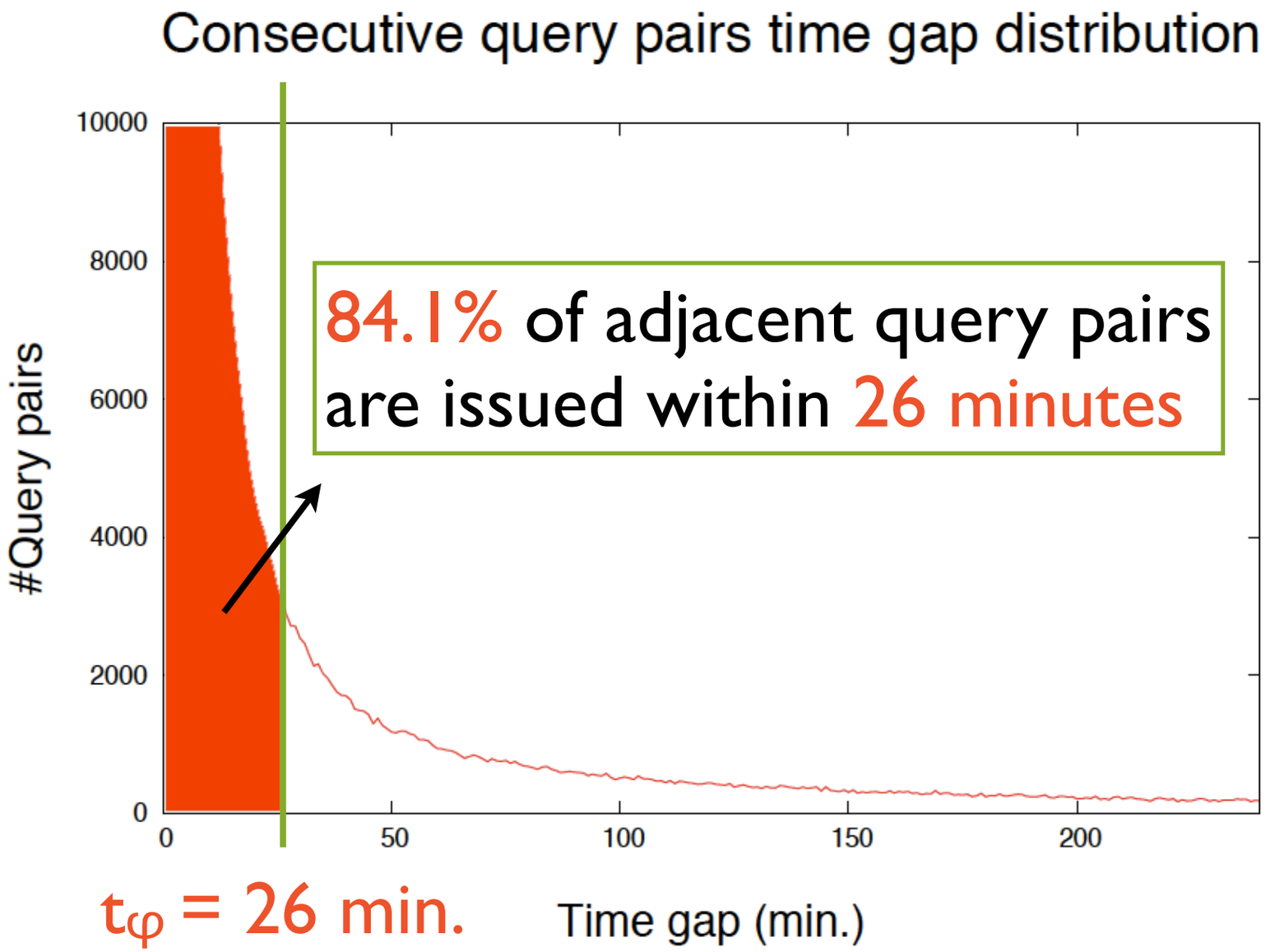
Sample Data Set

- ✓ 1-week collection
- ✓ ~100K queries
- ✓ 1,000 users
- ✓ removed empty queries
- ✓ removed "non-sense" queries
- ✓ removed stop-words
- ✓ applied Porter stemming algorithm

Data Analysis: query time gap



Data Analysis: query time gap



t_φ = 26 min.

Agenda

- Introduction
- Contributions
  - Query Log Analysis
  - Ground-truth Analysis
  - Approaching TSDP
- Experiments and Results
- Conclusions and Future Work

Ground-truth: construction

- Long-term sessions of sample data set are first split using the threshold t_φ devised before (i.e., 26 minutes)
  - obtaining several time-gap sessions

Ground-truth: construction

- Long-term sessions of sample data set are first split using the threshold t_φ devised before (i.e., 26 minutes)
  - obtaining several time-gap sessions
- Human annotators group queries that they claim to be task-related inside each time-gap session

Ground-truth: construction

- Long-term sessions of sample data set are first split using the threshold t_φ devised before (i.e., 26 minutes)
  - obtaining several time-gap sessions
- Human annotators group queries that they claim to be task-related inside each time-gap session
- Represents the true task-based partitioning manually built from actual WSE query log data

Ground-truth: construction

- Long-term sessions of sample data set are first split using the threshold t_φ devised before (i.e., 26 minutes)
  - obtaining several time-gap sessions
- Human annotators group queries that they claim to be task-related inside each time-gap session
- Represents the true task-based partitioning manually built from actual WSE query log data
- Useful both for statistical purposes and evaluation of automatic task-based session discovery methods

Ground-truth: statistics

- ✓ 2,004 queries
- ✓ 446 time-gap sessions
- ✓ 1,424 annotated queries
- ✓ 307 annotated time-gap sessions
- ✓ 554 detected task-based sessions

Ground-truth: statistics

- ✓ 4.49 avg. queries per time-gap session
- ✓ more than 70% time-gap session contains at most 5 queries

Ground-truth: statistics

- ✓ 4.49 avg. queries per time-gap session
- ✓ more than 70% time-gap session contains at most 5 queries

- ✓ 2.57 avg. queries per task
- ✓ ~75% tasks contains at most 3 queries

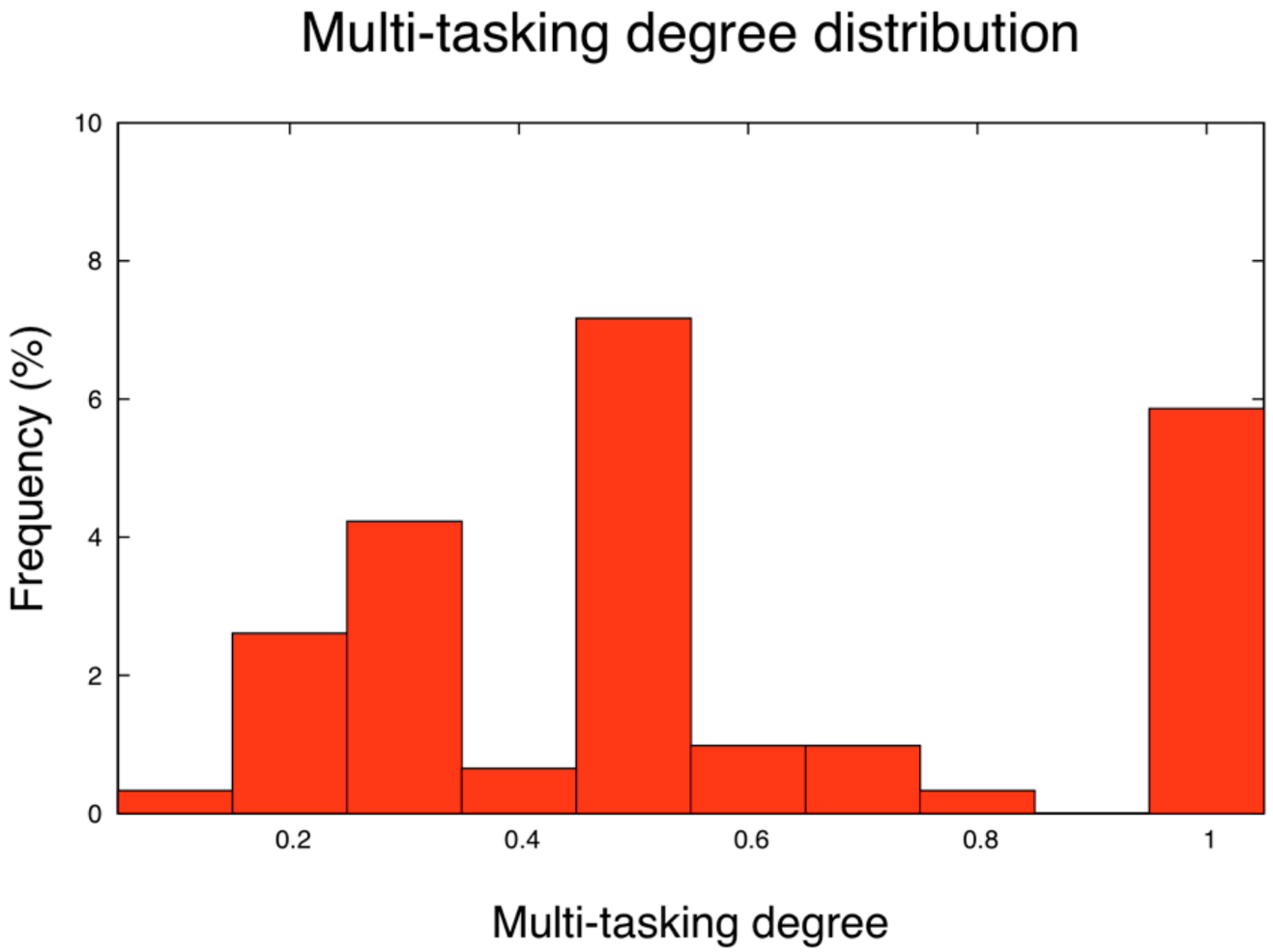
Ground-truth: statistics

- ✓ 4.49 avg. queries per time-gap session
- ✓ more than 70% time-gap session contains at most 5 queries

- ✓ 2.57 avg. queries per task
- ✓ ~75% tasks contains at most 3 queries

- ✓ 1.80 avg. task per time-gap session
- ✓ ~47% time-gap session contains more than one task (multi-tasking)
- ✓ 1,046 over 1,424 queries (i.e., ~74%) included in multi-tasking sessions

Ground-truth: statistics



- ✓ overlapping degree of multi-tasking sessions
- ✓ jump occurs whenever two queries of the same task are not originally adjacent
- ✓ ratio of task in a time-gap session that contains at least one jump

Agenda

- Introduction
- Contributions
  - Query Log Analysis
  - Ground-truth Analysis
  - Approaching TSDP
- Experiments and Results
- Conclusions and Future Work

TSDP: approaches

1) TimeSplitting-t

Description:

The idea is that if two consecutive queries are far away enough then they are also likely to be unrelated.

Two consecutive queries (q_i, q_{i+1}) are in the same task-based session if and only if their time submission gap is lower than a certain threshold t.

TSDP: approaches

1) TimeSplitting-t

Description:

The idea is that if two consecutive queries are far away enough then they are also likely to be unrelated.

Two consecutive queries (q_i, q_{i+1}) are in the same task-based session if and only if their time submission gap is lower than a certain threshold t.

PROs:

✓ ease of implementation

✓ $O(n)$ time complexity (linear in the number n of queries)

TSDP: approaches

1) TimeSplitting-t

Description:

The idea is that if two consecutive queries are far away enough then they are also likely to be unrelated.

Two consecutive queries (q_i, q_{i+1}) are in the same task-based session if and only if their time submission gap is lower than a certain threshold t.

PROs:

- ✓ ease of implementation
- ✓ $O(n)$ time complexity (linear in the number n of queries)

CONs:

- ✓ unable to deal with multi-tasking
- ✓ unawareness of other discriminating query features (e.g., lexical content)

TSDP: approaches

1) TimeSplitting-t

Description:

The idea is that if two consecutive queries are far away enough then they are also likely to be unrelated.

Two consecutive queries (q_i, q_{i+1}) are in the same task-based session if and only if their time submission gap is lower than a certain threshold t.

PROs:

- ✓ ease of implementation
- ✓ $O(n)$ time complexity (linear in the number n of queries)

CONs:

- ✓ unable to deal with multi-tasking
- ✓ unawareness of other discriminating query features (e.g., lexical content)

Methods: TS-5, TS-15, TS-26, etc.

TSDP: approaches

1) TimeSplitting-t

Description:

The idea is that if two consecutive queries are far away enough then they are also likely to be unrelated.

Two consecutive queries (q_i, q_{i+1}) are in the same task-based session if and only if their time submission gap is lower than a certain threshold t.

PROs:

- ✓ ease of implementation
- ✓ O(n) time complexity (linear in the number n of queries)

CONs:

- ✓ unable to deal with multi-tasking
- ✓ unawareness of other discriminating query features (e.g., lexical content)

Methods: TS-5, TS-15, TS-26, etc.

2) QueryClustering-m

Description:

Queries are grouped using clustering algorithms, which exploit several query features. Clustering algorithms assembly such features using two different distance functions for computing query-pair similarity.

Two queries (q_i, q_j) are in the same task-based session if and only if they are in the same cluster.

TSDP: approaches

1) TimeSplitting-t

Description:

The idea is that if two consecutive queries are far away enough then they are also likely to be unrelated.

Two consecutive queries (q_i, q_{i+1}) are in the same task-based session if and only if their time submission gap is lower than a certain threshold t.

PROs:

- ✓ ease of implementation
- ✓ O(n) time complexity (linear in the number n of queries)

CONs:

- ✓ unable to deal with multi-tasking
- ✓ unawareness of other discriminating query features (e.g., lexical content)

Methods: TS-5, TS-15, TS-26, etc.

2) QueryClustering-m

Description:

Queries are grouped using clustering algorithms, which exploit several query features. Clustering algorithms assembly such features using two different distance functions for computing query-pair similarity.

Two queries (q_i, q_j) are in the same task-based session if and only if they are in the same cluster.

PROs:

- ✓ able to detect multi-tasking sessions
- ✓ able to deal with "noisy queries" (i.e., outliers)

TSDP: approaches

1) TimeSplitting-t

Description:

The idea is that if two consecutive queries are far away enough then they are also likely to be unrelated.

Two consecutive queries (q_i, q_{i+1}) are in the same task-based session if and only if their time submission gap is lower than a certain threshold t.

PROs:

- ✓ ease of implementation
- ✓ O(n) time complexity (linear in the number n of queries)

CONs:

- ✓ unable to deal with multi-tasking
- ✓ unawareness of other discriminating query features (e.g., lexical content)

Methods: TS-5, TS-15, TS-26, etc.

2) QueryClustering-m

Description:

Queries are grouped using clustering algorithms, which exploit several query features. Clustering algorithms assembly such features using two different distance functions for computing query-pair similarity.

Two queries (q_i, q_j) are in the same task-based session if and only if they are in the same cluster.

PROs:

- ✓ able to detect multi-tasking sessions
- ✓ able to deal with "noisy queries" (i.e., outliers)

CONs:

- ✓ $O(n^2)$ time complexity (i.e. quadratic in the number n of queries due to all-pairs-similarity computational step)

TSDP: approaches

1) TimeSplitting-t

Description:

The idea is that if two consecutive queries are far away enough then they are also likely to be unrelated.

Two consecutive queries (q_i, q_{i+1}) are in the same task-based session if and only if their time submission gap is lower than a certain threshold t.

PROs:

- ✓ ease of implementation
- ✓ O(n) time complexity (linear in the number n of queries)

CONs:

- ✓ unable to deal with multi-tasking
- ✓ unawareness of other discriminating query features (e.g., lexical content)

Methods: TS-5, TS-15, TS-26, etc.

2) QueryClustering-m

Description:

Queries are grouped using clustering algorithms, which exploit several query features. Clustering algorithms assembly such features using two different distance functions for computing query-pair similarity.

Two queries (q_i, q_j) are in the same task-based session if and only if they are in the same cluster.

PROs:

- ✓ able to detect multi-tasking sessions
- ✓ able to deal with "noisy queries" (i.e., outliers)

CONs:

- ✓ $O(n^2)$ time complexity (i.e. quadratic in the number n of queries due to all-pairs-similarity computational step)

Methods: QC-MEANS, QC-SCAN, QC-WCC, and QC-HTC

Query Features

Content-based ($\mu_{content}$)

- ✓ two queries (q_i, q_j) sharing common terms are likely related
- ✓ $\mu_{jaccard}$: Jaccard index on query character 3-grams

$$\mu_{jaccard}(q_1, q_2) = 1 - \frac{|T(q_1) \cap T(q_2)|}{|T(q_1) \cup T(q_2)|}$$

- ✓ $\mu_{levenshtein}$: normalized Levenshtein distance

$$\mu_{content}(q_1, q_2) = \frac{(\mu_{jaccard} + \mu_{levenshtein})}{2}$$

Query Features

Content-based ($\mu_{content}$)

- ✓ two queries (q_i, q_j) sharing common terms are likely related
- ✓ $\mu_{jaccard}$: Jaccard index on query character 3-grams

$$\mu_{jaccard}(q_1, q_2) = 1 - \frac{|T(q_1) \cap T(q_2)|}{|T(q_1) \cup T(q_2)|}$$

- ✓ $\mu_{levenshtein}$: normalized Levenshtein distance

$$\mu_{content}(q_1, q_2) = \frac{(\mu_{jaccard} + \mu_{levenshtein})}{2}$$

Semantic-based ($\mu_{semantic}$)

- ✓ using Wikipedia and Wiktionary for "expanding" a query q
- ✓ "wikification" of q using vector-space model

$$\vec{C}(t) = (c_1, c_2, \ldots, c_W) \qquad \vec{C}(q) = \sum_{t \in q} \vec{C}(t)$$

- ✓ relatedness between (q_i, q_j) computed using cosine-similarity

$$rel(q_1, q_2) = \frac{\vec{C}(q_1) \cdot \vec{C}(q_2)}{|\vec{C}(q_1)||\vec{C}(q_1)|}$$

$$\mu_{wikification}(q_1, q_2) = 1 - rel(q_1, q_2)$$

$$\mu_{semantic}(q_1, q_2) = min(\mu_{wiktionary}, \mu_{wikipedia})$$

Distance Functions: μ_1 vs. μ_2

✓ Convex combination μ_1

$$\mu_1 = \alpha \cdot \mu_{content} + (1 - \alpha) \cdot \mu_{semantic}$$

✓ Conditional formula μ_2

Idea: if two queries are close in term of lexical content, the semantic expansion could be unhelpful. Vice-versa, nothing can be said when queries do not share any content feature

$$\mu_2 = \begin{cases} \mu_{content} & \text{if } \mu_{content} < \mathbf{t} \\ \min(\mu_{content}, \mathbf{b} \cdot \mu_{semantic}) & \text{otherwise.} \end{cases}$$

✓ Both μ_1 and μ_2 rely on the estimation of some parameters, i.e., α, t, and b

✓ Use ground-truth for tuning parameters

QC-WCC

- Models each time-gap session φ as a complete weighted undirected graph $G_\varphi = (V, E, w)$

QC-WCC

- Models each time-gap session φ as a complete weighted undirected graph $G_\varphi = (V, E, w)$
  - set of nodes V are the queries in φ

QC-WCC

- Models each time-gap session φ as a complete weighted undirected graph $G_\varphi = (V, E, w)$
  - set of nodes V are the queries in φ
  - set of edges E are weighted by the similarity of the corresponding nodes

QC-WCC

- Models each time-gap session φ as a complete weighted undirected graph $G_\varphi = (V, E, w)$
  - set of nodes V are the queries in φ
  - set of edges E are weighted by the similarity of the corresponding nodes
- Drop weak edges, i.e., with low similarity, assuming the corresponding queries are not related and obtaining G'_φ

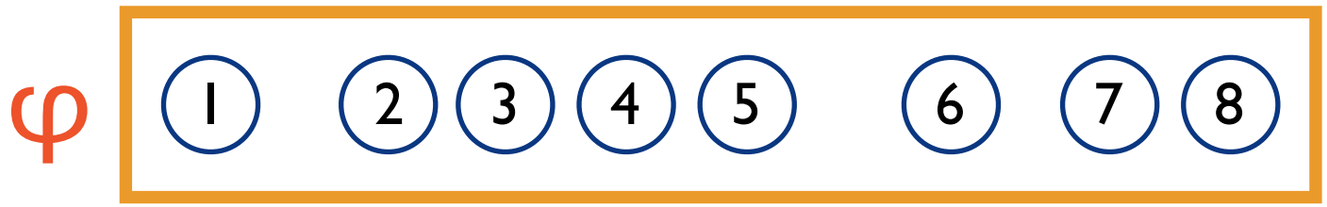
QC-WCC

- Models each time-gap session φ as a complete weighted undirected graph $G_\varphi = (V, E, w)$
  - set of nodes V are the queries in φ
  - set of edges E are weighted by the similarity of the corresponding nodes
- Drop weak edges, i.e., with low similarity, assuming the corresponding queries are not related and obtaining G'_φ
- Clusters are built on the basis of strong edges by finding all the connected components of the pruned graph G'_φ

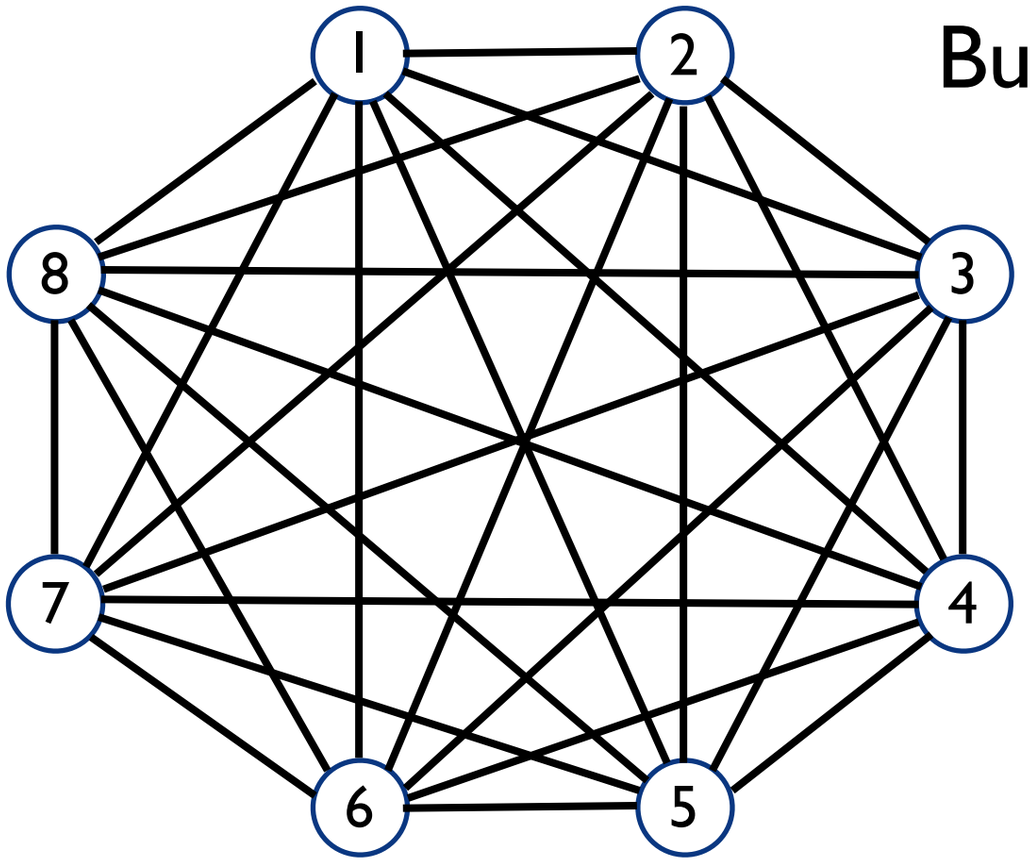
QC-WCC

- Models each time-gap session φ as a complete weighted undirected graph $G_\varphi = (V, E, w)$
  - set of nodes V are the queries in φ
  - set of edges E are weighted by the similarity of the corresponding nodes
- Drop weak edges, i.e., with low similarity, assuming the corresponding queries are not related and obtaining G'_φ
- Clusters are built on the basis of strong edges by finding all the connected components of the pruned graph G'_φ
- $O(n^2)$ time complexity where $n = |V|$

QC-WCC

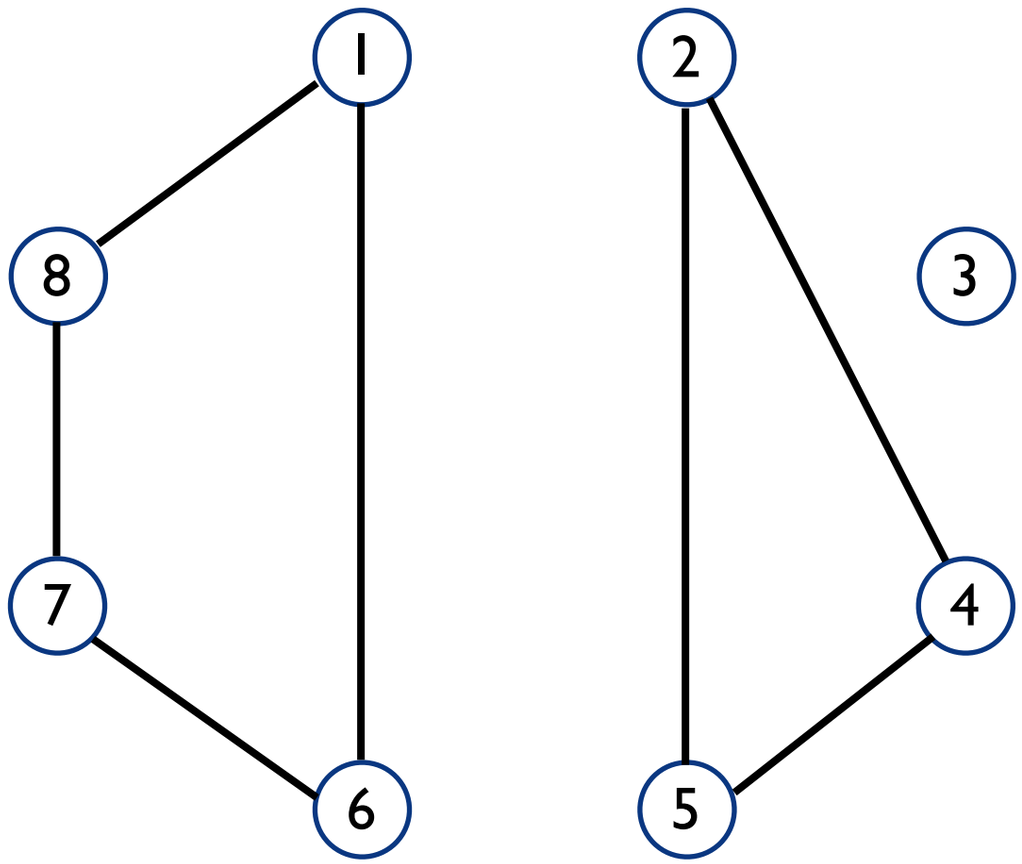


QC-WCC



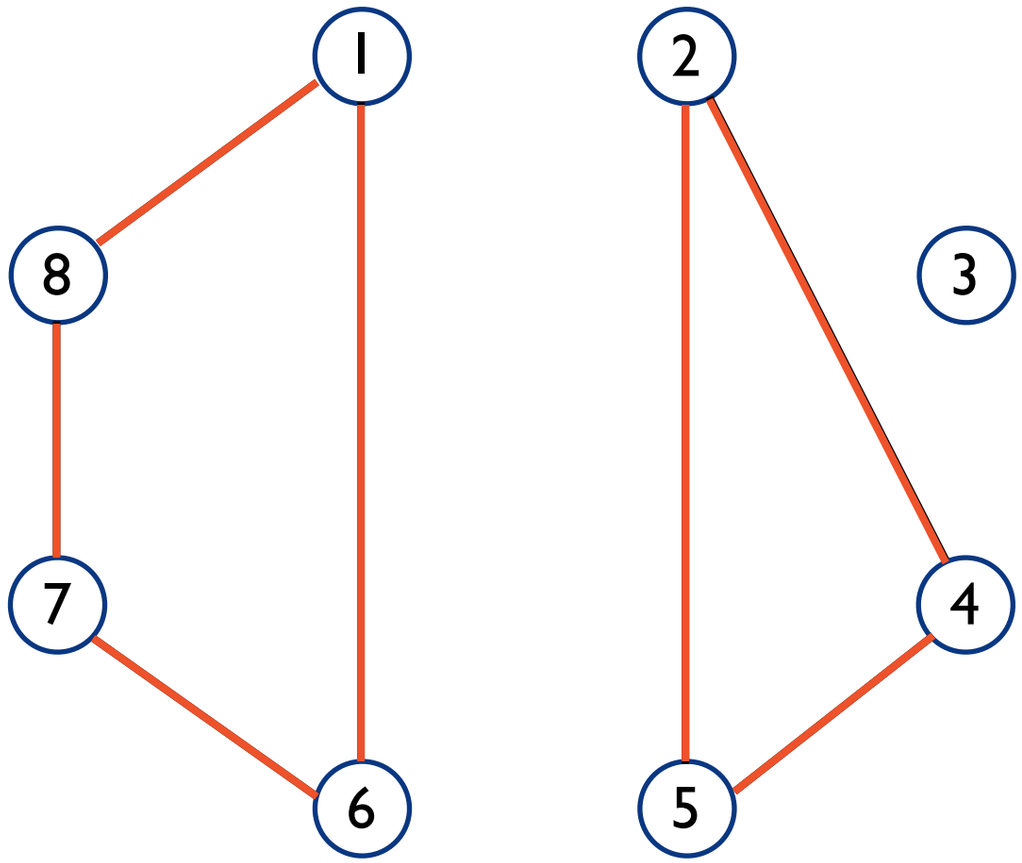
Build similarity graph G_φ

QC-WCC



Drop "weak edges"

QC-WCC



QC-HTC

- Variation of QC-WCC based on head-tail components

QC-HTC

- Variation of QC-WCC based on head-tail components
- Does not need to compute the full similarity graph

QC-HTC

- Variation of QC-WCC based on head-tail components
- Does not need to compute the full similarity graph
- Exploits the sequentiality of query submissions to reduce the number of similarity computations

QC-HTC

- Variation of QC-WCC based on head-tail components
- Does not need to compute the full similarity graph
- Exploits the sequentiality of query submissions to reduce the number of similarity computations
- Performs 2 steps:
  1. sequential clustering
  2. merging

QC-HTC: sequential clustering

- Partition each time-gap session into sequential clusters containing only queries issued in a row

QC-HTC: sequential clustering

- Partition each time-gap session into sequential clusters containing only queries issued in a row
- Each query in every sequential cluster has to be "similar enough" to the chronologically next one

QC-HTC: sequential clustering

- Partition each time-gap session into sequential clusters containing only queries issued in a row
- Each query in every sequential cluster has to be "similar enough" to the chronologically next one
- Need to compute only the similarity between one query and the next in the original data

QC-HTC: merging

- Merge together related sequential clusters due to multi-tasking

QC-HTC: merging

- Merge together related sequential clusters due to multi-tasking
- Hyp: a cluster is represented by its chronologically-first and last queries, i.e., head and tail, respectively

QC-HTC: merging

- Merge together related sequential clusters due to multi-tasking
- Hyp: a cluster is represented by its chronologically-first and last queries, i.e., head and tail, respectively

- Given two sequential clusters c_i, c_j and h_i, t_i, and h_j, t_j, their corresponding head and tail queries the similarity $s(c_i, c_j)$ is computed as follow:

QC-HTC: merging

- Merge together related sequential clusters due to multi-tasking
- Hyp: a cluster is represented by its chronologically-first and last queries, i.e., head and tail, respectively

- Given two sequential clusters c_i, c_j and h_i, t_i, and h_j, t_j, their corresponding head and tail queries the similarity $s(c_i, c_j)$ is computed as follow:

$$s(c_i, c_j) = \min w(e(q_i, q_j)) \text{ s.t. } q_i \in \{h_i, t_i\} \text{ and } q_j \in \{h_j, t_j\}$$

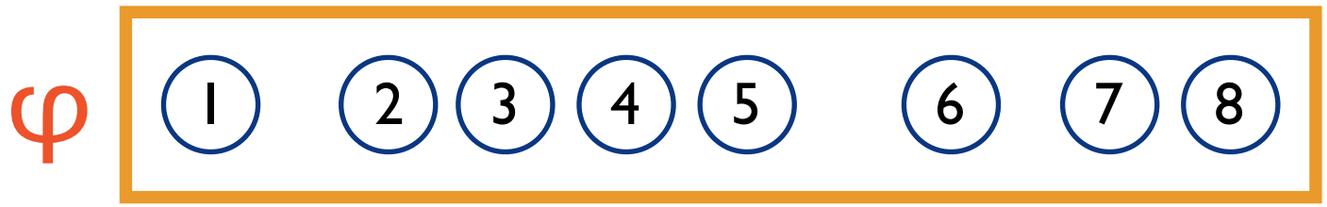
QC-HTC: merging

- Merge together related sequential clusters due to multi-tasking
- Hyp: a cluster is represented by its chronologically-first and last queries, i.e., head and tail, respectively
- Given two sequential clusters c_i, c_j and h_i, t_i, and h_j, t_j, their corresponding head and tail queries the similarity $s(c_i, c_j)$ is computed as follow:

$$s(c_i, c_j) = \min w(e(q_i, q_j)) \text{ s.t. } q_i \in \{h_i, t_i\} \text{ and } q_j \in \{h_j, t_j\}$$

  - c_i and c_j are merged as long as $s(c_i, c_j) > \eta$
  - h_i, t_i and h_j, t_j are updated consequently

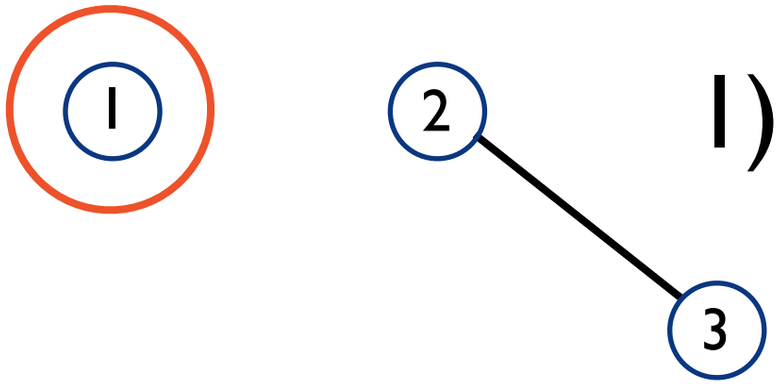
QC-HTC



QC-HTC

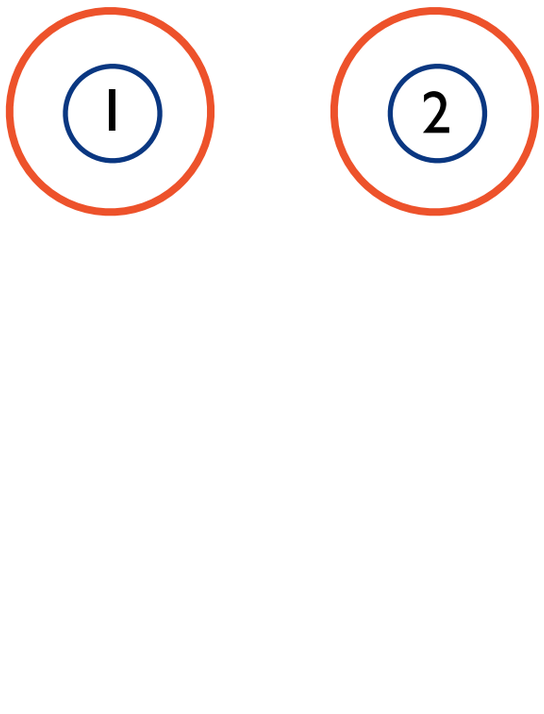
1) Sequential Clustering

QC-HTC



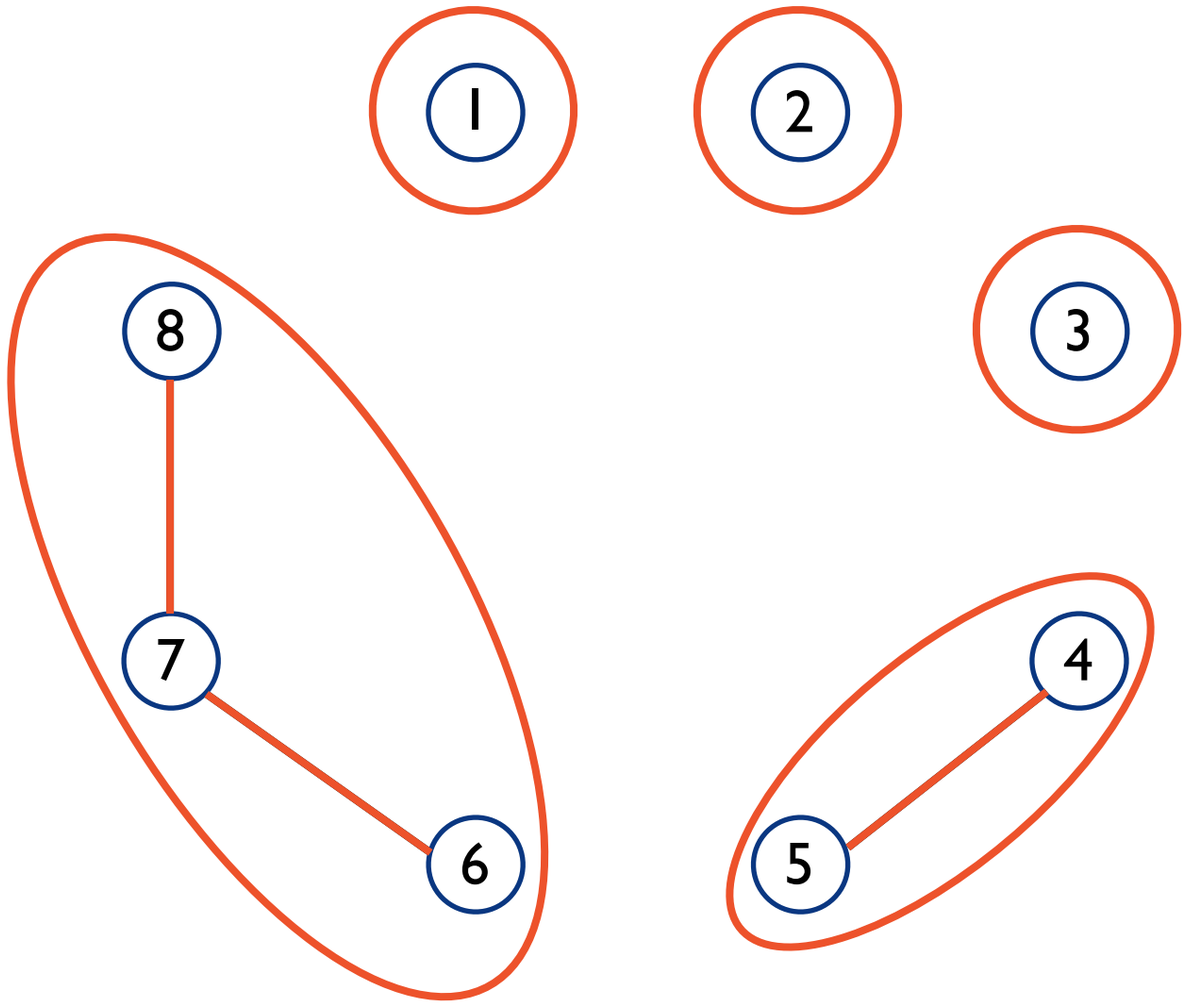
1) Sequential Clustering

QC-HTC

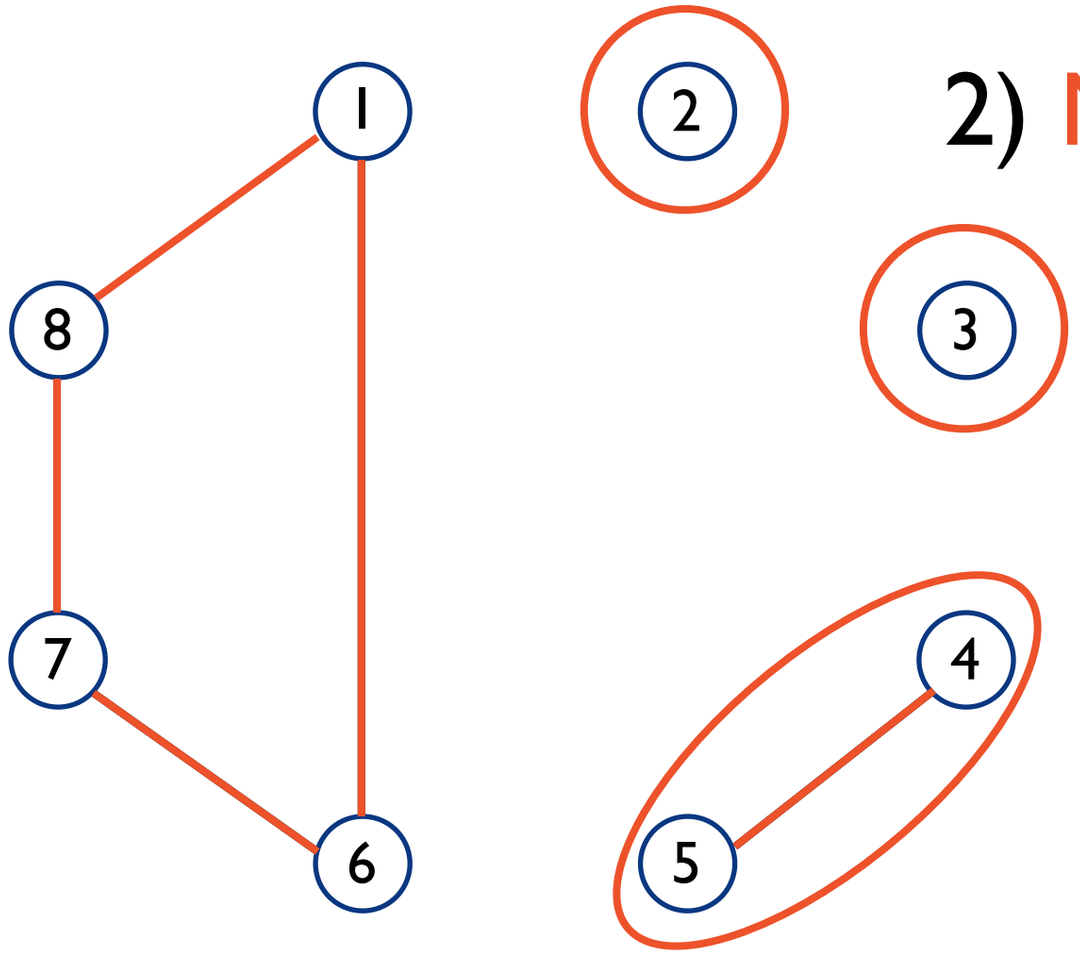


1) Sequential Clustering

QC-HTC

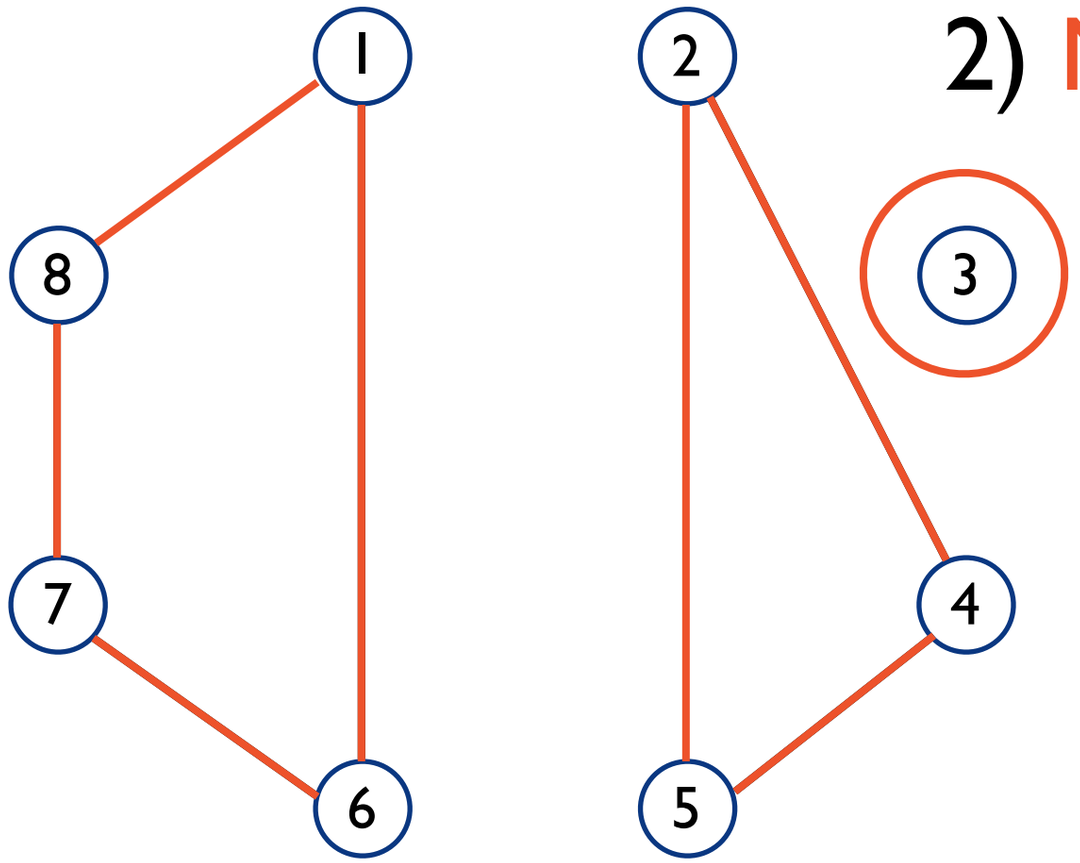


QC-HTC



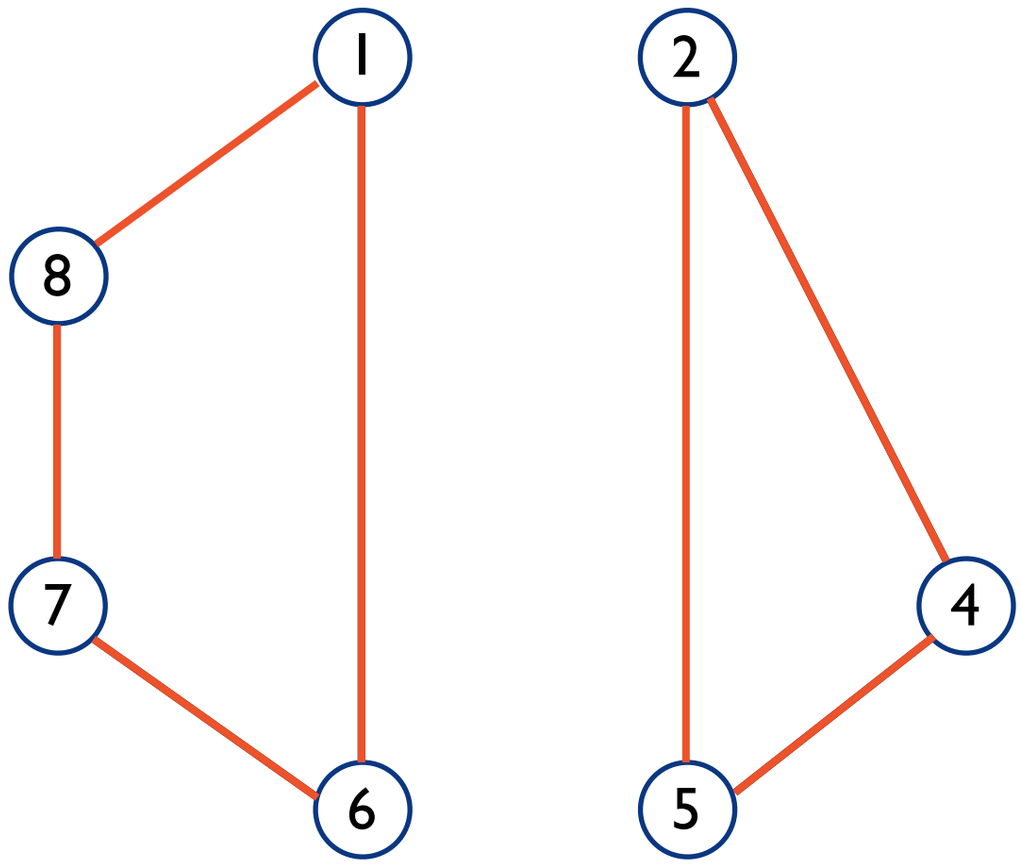
2) Merging

QC-HTC



2) Merging

QC-HTC



QC-HTC: time complexity

- In the first step the algorithm computes the similarity only between one query and the next in the original data
  - O(n) where n is the size of the time-gap session

QC-HTC: time complexity

- In the first step the algorithm computes the similarity only between one query and the next in the original data
  - $O(n)$ where n is the size of the time-gap session
- In the second step the algorithm computes the pairwise similarity between each sequential cluster
  - $O(k^2)$ where k is the number of sequential clusters
  - if $k = \beta \cdot n$ with $0<\beta\leq 1$ then time complexity is $O(\beta^2 \cdot n^2)$
  - e.g. $\beta = 1/2 \Rightarrow O(n^2/4) \Rightarrow$ 4 times better than QC-WCC

Agenda

- Introduction
- Contributions
- Experiments and Results
- Conclusions and Future Work

Setup

- Run and compare all the proposed approaches with:

Setup

- Run and compare all the proposed approaches with:
  - TS-26: time-splitting technique (baseline)

Setup

- Run and compare all the proposed approaches with:
  - TS-26: time-splitting technique (baseline)
  - QFG: session extraction method based on the query-flow graph model (state of the art)

Evaluation

- Measure the degree of correspondence between true tasks, i.e., manually-extracted ground-truth, and predicted tasks, i.e., output by algorithms

Evaluation

- Measure the degree of correspondence between true tasks, i.e., manually-extracted ground-truth, and predicted tasks, i.e., output by algorithms

a) F-MEASURE

✓ evaluates the extent to which a predicted task contains only and all the queries of a true task

✓ combines p(i, j) and r(i, j) the precision and recall of task i w.r.t. class j

$$F(i,j) = \frac{2 \times p(i,j) \times r(i,j)}{p(i,j) + r(i,j)}$$

Evaluation

- Measure the degree of correspondence between true tasks, i.e., manually-extracted ground-truth, and predicted tasks, i.e., output by algorithms

a) F-MEASURE

✓ evaluates the extent to which a predicted task contains only and all the queries of a true task

✓ combines p(i, j) and r(i, j) the precision and recall of task i w.r.t. class j

$$F(i,j) = \frac{2 \times p(i,j) \times r(i,j)}{p(i,j) + r(i,j)}$$

b) RAND

✓ pairs of queries instead of singleton

✓ f_{00}, f_{01}, f_{10}, f_{11}

$$R = \frac{f_{00} + f_{11}}{f_{00} + f_{01} + f_{10} + f_{11}}$$

Evaluation

- Measure the degree of correspondence between true tasks, i.e., manually-extracted ground-truth, and predicted tasks, i.e., output by algorithms

a) F-MEASURE

✓ evaluates the extent to which a predicted task contains only and all the queries of a true task

✓ combines p(i, j) and r(i, j) the precision and recall of task i w.r.t. class j

$$F(i,j) = \frac{2 \times p(i,j) \times r(i,j)}{p(i,j) + r(i,j)}$$

b) RAND

✓ pairs of queries instead of singleton

✓ f_{00}, f_{01}, f_{10}, f_{11}

$$R = \frac{f_{00} + f_{11}}{f_{00} + f_{01} + f_{10} + f_{11}}$$

c) JACCARD

✓ pairs of queries instead of singleton

✓ f_{01}, f_{10}, f_{11}

$$J = \frac{f_{11}}{f_{01} + f_{10} + f_{11}}$$

Results: TS-t

Table 1: TS-5, TS-15, **and** TS-26.

	F-measure	Rand	Jaccard
TS-5	0.28	**0.75**	0.03
TS-15	0.28	0.71	0.08
TS-26	**0.65**	0.34	**0.34**

- 3 time thresholds used: 5, 15, and 26 minutes

Results: TS-t

Table 1: TS-5, TS-15, **and** TS-26.

	F-measure	Rand	Jaccard
TS-5	0.28	**0.75**	0.03
TS-15	0.28	0.71	0.08
TS-26	**0.65**	0.34	**0.34**

- 3 time thresholds used: 5, 15, and 26 minutes
- Note: TS-26 was used for splitting sample data set
  - task-based sessions == time-gap sessions

Results: QFG

Table 2: QFG: varying the threshold η.

	η	F-measure	Rand	Jaccard
QFG	0.1	0.68	0.47	0.36
	0.2	0.68	0.49	0.36
	0.3	0.69	0.51	0.37
	0.4	0.70	0.55	0.38
	0.5	0.71	0.59	0.38
	0.6	0.74	0.65	0.39
	0.7	**0.77**	**0.71**	**0.40**
	0.8	0.77	0.71	0.40
	0.9	0.77	0.71	0.40

- ✓ trained on a segment of our sample data set
- ✓ best results using η = 0.7
- ✓ vs. baseline:
  - +16% F-measure
  - +52% Rand
  - +15% Jaccard

Results: QC-WCC

Table 5: QC-WCC: μ_1 **vs.** μ_2 **varying the threshold** η.

QC-WCC μ_1 ($\alpha = 0.5$)			
η	F-measure	Rand	Jaccard
0.1	0.78	0.71	0.42
0.2	**0.81**	**0.78**	**0.43**
0.3	0.79	0.77	0.37
0.4	0.75	0.73	0.27
0.5	0.72	0.71	0.20
0.6	0.75	0.70	0.14
0.7	0.74	0.69	0.11
0.8	0.74	0.68	0.07
0.9	0.72	0.67	0.04

QC-WCC $\mu_2 (t = 0.5, b = 4)$			
η	F-measure	Rand	Jaccard
0.1	0.67	0.45	0.33
0.2	0.78	0.71	0.42
0.3	**0.81**	**0.78**	**0.44**
0.4	0.81	0.78	0.41
0.5	0.80	0.77	0.37
0.6	0.78	0.75	0.32
0.7	0.75	0.73	0.23
0.8	0.71	0.70	0.15
0.9	0.69	0.68	0.08

- ✓ best results using μ_2 and $\eta = 0.3$
- ✓ vs. baseline:
  - +20% F-measure
  - +56% Rand
  - +23% Jaccard
- ✓ vs. QFG:
  - +5% F-measure
  - +9% Rand
  - +10% Jaccard

Results: QC-HTC

Table 6: QC-HTC: μ_1 **vs.** μ_2 **varying the threshold** η.

QC-HTC μ_1 ($\alpha = 0.5$)			
η	F-measure	Rand	Jaccard
0.1	0.78	0.72	0.41
0.2	**0.80**	**0.78**	**0.41**
0.3	0.78	0.76	0.35
0.4	0.75	0.73	0.25
0.5	0.73	0.70	0.18
0.6	0.75	0.70	0.13
0.7	0.74	0.69	0.10
0.8	0.74	0.68	0.06
0.9	0.72	0.67	0.03

QC-HTC μ_2 ($t = 0.5, b = 4$)			
η	F-measure	Rand	Jaccard
0.1	0.68	0.56	0.32
0.2	0.78	0.73	0.41
0.3	**0.80**	**0.78**	**0.43**
0.4	0.80	0.77	0.38
0.5	0.78	0.76	0.34
0.6	0.77	0.74	0.30
0.7	0.74	0.72	0.21
0.8	0.71	0.70	0.14
0.9	0.68	0.67	0.07

- ✓ best results using μ_2 and $\eta = 0.3$
- ✓ vs. baseline:
  - +19% F-measure
  - +56% Rand
  - +21% Jaccard
- ✓ vs. QFG:
  - +4% F-measure
  - +9% Rand
  - +8% Jaccard

Results: best

Table 7: Best results obtained with each method.

	F-measure	Rand	Jaccard
TS-26 (*baseline*)	0.65	0.34	0.34
QFG $_{best}$ (*state of the art*)	0.77	0.71	0.40
QC-MEANS $_{best}$	0.72	0.74	0.27
QC-SCAN $_{best}$	0.77	0.71	0.19
QC-WCC $_{best}$	**0.81**	**0.78**	**0.44**
QC-HTC $_{best}$	0.80	0.78	0.43

Results: Wiki impact

Table 8: The impact of Wikipedia: μ_1 vs. μ_2

QC-HTC μ_1 $(\alpha = 1)$		QC-HTC μ_2 $(0.5, 4)$	
Query ID	Query String	Query ID	Query String
		63	los cabos
		64	cancun
65	hurricane wilma	65	hurricane wilma
68	hurricane wilma	68	hurricane wilma

- Benefit of using Wikipedia instead of only lexical content when computing query distance function

Results: Wiki impact

Table 8: The impact of Wikipedia: μ_1 vs. μ_2

QC-HTC μ_1 $(\alpha = 1)$		QC-HTC μ_2 $(0.5, 4)$	
Query ID	Query String	Query ID	Query String
		63	los cabos
		64	cancun
65	hurricane wilma	65	hurricane wilma
68	hurricane wilma	68	hurricane wilma

- Benefit of using Wikipedia instead of only lexical content when computing query distance function
- Capturing other two queries that are lexically different but somehow "semantically" similar

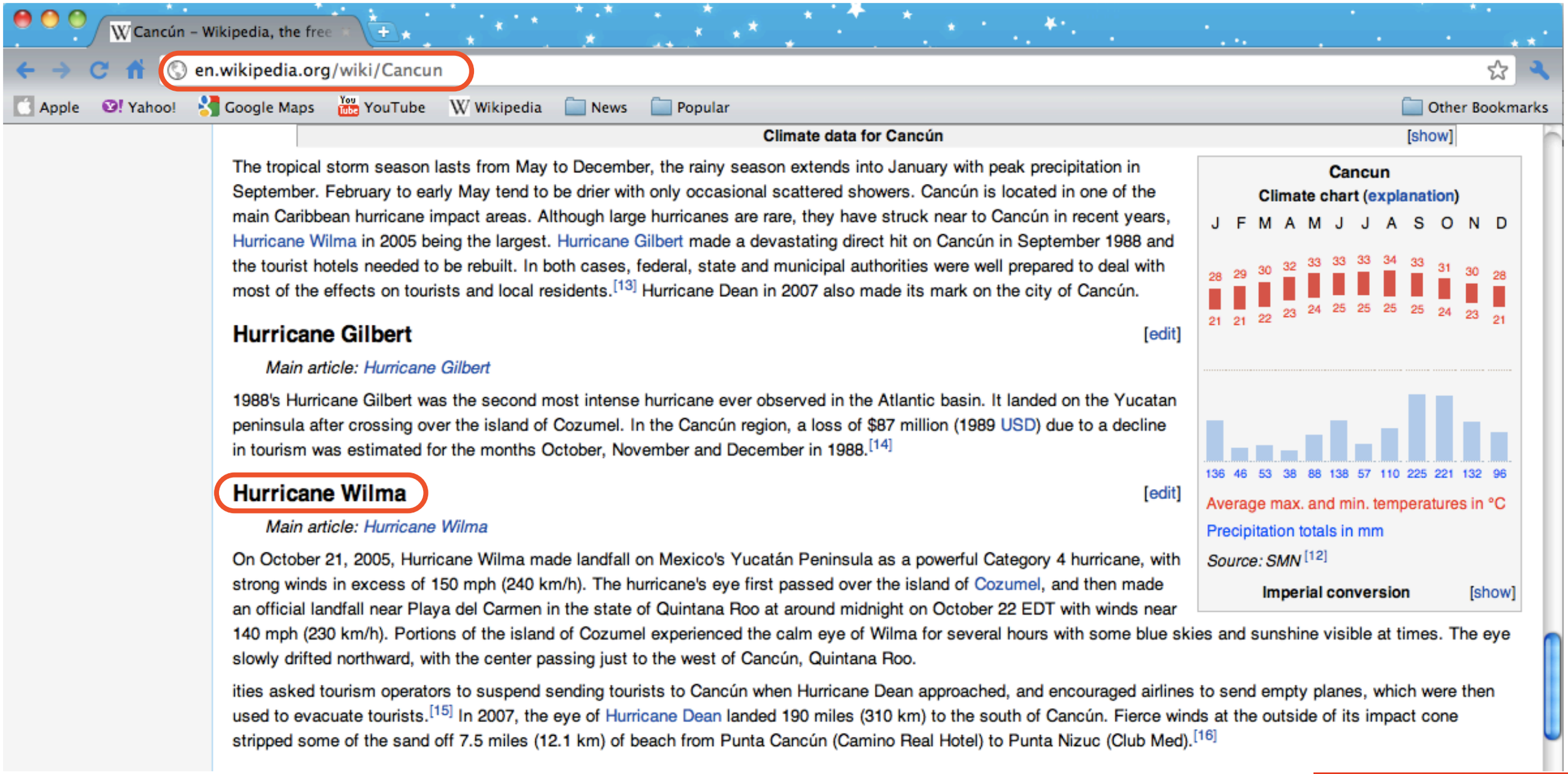
Results: Wiki impact

Table 8: The impact of Wikipedia: μ_1 vs. μ_2

QC-HTC μ_1 $(\alpha = 1)$		QC-HTC μ_2 (0.5, 4)	
Query ID	Query String	Query ID	Query String
		63	los cabos
		64	cancun
65	hurricane wilma	65	hurricane wilma
68	hurricane wilma	68	hurricane wilma

- Benefit of using Wikipedia instead of only lexical content when computing query distance function
- Capturing other two queries that are lexically different but somehow "semantically" similar
- Try going here: http://en.wikipedia.org/wiki/Cancun

Results: Wiki impact



Agenda

- Introduction
- Contributions
- Experiments and Results
- Conclusions and Future Work

Conclusions

- Introduced the Task-based Session Discovery Problem
  - from a WSE log of user activities extract several sets of queries which are all related to the same task

Conclusions

- Introduced the Task-based Session Discovery Problem
  - from a WSE log of user activities extract several sets of queries which are all related to the same task

- Compared clustering solutions exploiting two distance functions based on query content and semantic expansion (i.e., Wiktionary and Wikipedia)

Conclusions

- Introduced the Task-based Session Discovery Problem
  - from a WSE log of user activities extract several sets of queries which are all related to the same task

- Compared clustering solutions exploiting two distance functions based on query content and semantic expansion (i.e., Wiktionary and Wikipedia)

- Proposed novel graph-based heuristic QC-HTC, lighter than QC-WCC, outperforming other methods in terms of F-measure, Rand and Jaccard index

Future Work

- Why should we stop here?

Future Work

- Why should we stop here?
- Once discovered, smaller tasks might be part of larger and more complex tasks

Future Work

- Why should we stop here?
- Once discovered, smaller tasks might be part of larger and more complex tasks
- The task "fly to Hong Kong" might be a step of a larger task, e.g., "holidays in Hong Kong", which in turn could involve several other tasks...

Vision

- Make Web Search Engine the "universal driver" for executing our daily activities on the Web

Vision

- Make Web Search Engine the "universal driver" for executing our daily activities on the Web
- Once user types in a query, WSE should "infer the tasks" user aims to perform (if any) ⇒ serendipity!

Vision

- Make Web Search Engine the "universal driver" for executing our daily activities on the Web
- Once user types in a query, WSE should "infer the tasks" user aims to perform (if any) ⇒ serendipity!
- Results should be no longer only list of plain links but also tasks, either simple and complex

Vision

- Make Web Search Engine the "universal driver" for executing our daily activities on the Web
- Once user types in a query, WSE should "infer the tasks" user aims to perform (if any) ⇒ serendipity!
- Results should be no longer only list of plain links but also tasks, either simple and complex
- Recommendation of queries and/or Web pages both intra- and inter-task

task vs. query recommendation

References

[1] Silverstein, Marais, Henzinger, and Moricz. "*Analysis of a very large web search engine query log*". In SIGIR Forum, 1999

[2] He and Göker. "*Detecting session boundaries from web user logs*". In BCS-IRSG, 2000

[3] Radlinski and Joachims. "*Query chains: Learning to rank from implicit feedback*". In KDD '05

[4] Jansen and Spink. "*How are we searching the world wide web?: a comparison of nine search engine transaction logs*". In IPM, 2006

[5] Lau and Horvitz. "*Patterns of search: Analyzing and modeling web query refinement*". In UM '99

[6] He and Harper. "*Combining evidence for automatic web session identification*". In IPM, 2002

[7] Ozmutlu and Çavdur. "*Application of automatic topic identification on excite web search engine data logs*". In IPM, 2005

[8] Shen, Tan, and Zhai. "*Implicit user modeling for personalized search*". In CIKM '05

[9] Boldi, Bonchi, Castillo, Donato, Gionis, and Vigna. "*The query-flow graph: model and applications*". In CIKM '08

[10] Jones and Klinkner. "*Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs*". In CIKM '08

[11] MacQueen. "*Some methods for classification and analysis of multivariate observations*". In BSMSP, 1967

[12] Ester, Kriegel, Sander, and Xu. "*A density-based algorithm for discovering clusters in large spatial databases with noise*". In KDD '96

Thank You!

Questions?