

Towards Effective and Efficient Behavior-based Trust Models

Klemens Böhm

Universität Karlsruhe (TH)

Motivation: Grid Computing in Particle Physics

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

- Physicists have designed and implemented services specific to particle physics (data analysis, computations, etc.).
- In general, physicists are willing to let others use their services.
- A service typically runs at the institution where it has been implemented.
- However, resources are limited etc. Typically, one cannot/does not want to process all service requests.
- Letting others pay for services is not acceptable
 - for cultural reasons, and
 - rigid specification of service characteristics would be necessary.

Motivation, continued

- Different users have different policies.
Highly subjective issue.
- Wanted: Language to specify when a particular user may consume the service.
- Behavior-based:
 - Decision depends on previous behavior of the user.
 - Useful in settings that are anonymous.
 - Requires that information on previous behavior of the user is available.
- Aka. *trust policies/trust policy languages*.

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Behavior-based Trust Policies (1)

- Example policies:
 - Alice: "I let someone use my resources (i.e., I trust him) if the average feedback about him is positive."
 - Bob: "I will provide service X for someone if there is no negative feedback about him within the last 24h."
 - Carol : "I will only interact with someone if the k most reputable entities recommend him."
 - Dave: "I only perform the services for others if their performance regarding complex tasks has been satisfactorily."

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Behavior-based Trust Policies (2)

- Not only service providers may use such policies, but also consumers of services.
(E.g., "I only want to use services of providers without any negative feedback about them.")

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

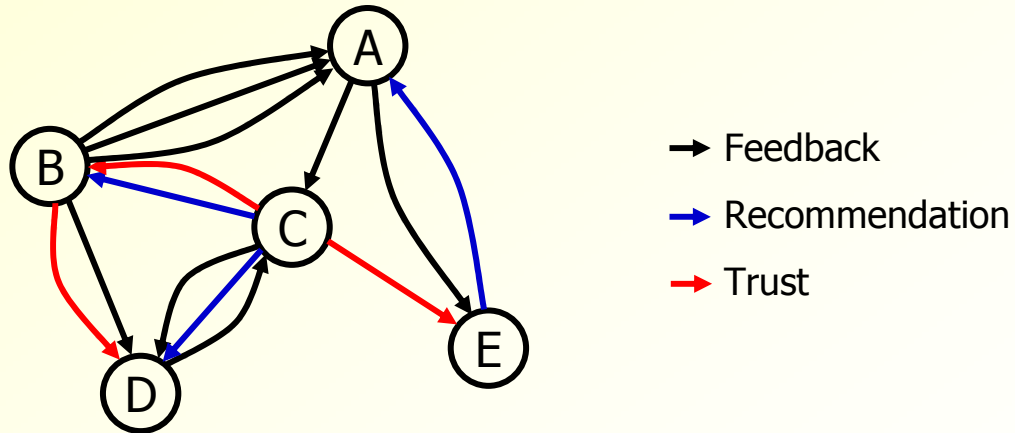
What Can We Learn from the Examples? (1)

- Requirement 1:
Trust policies may require complex operations, e.g., aggregation ('average feedback').
- Requirement 2:
Adequate representation of knowledge that describes behavior of participants sought (*behavior-specific knowledge*).
 - Different types of behavior-specific knowledge: feedback, reputation, recommendation, trust.
 - Various aspects of behavior-specific knowledge considered (e.g., context, age of knowledge, etc.).

What Can We Learn from the Examples? (2)

- Representation of knowledge as directed graph $G(V,E)$
 - V ...set of participants
 - E ...set of edges based on behavior-specific knowledge

- Example:



- ➔ Application of graph algorithms to find trustworthy partners e.g., EigenTrust (Schlosser et al., 2003), PageRank (Brin and Page, 1996). *Centrality*.

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Motivation, Continued

- Research issues:
 - Define a language to formulate trust policies.
 - Efficient evaluation of policies.
 - Effectiveness issues.

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Contributions

- Identify characteristics of/classify behavior-based trust models from literature. Definition of underlying concepts.
- Definition of query algebra for trust. Demonstrated its appropriateness.
- Experiments:
 - Efficiency and effectiveness of various trust policies (relying on centrality measures).
 - How does preprocessing of underlying data affect the effectiveness of centrality measures?
 - Which trust policies are used in reality? (ongoing work)

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Agenda

- Introduction
- Terminology
- Centrality
- Query Algebra for Trust
 - Idea
 - Conventional Extensions
 - Centrality Operator
- Experiments
 - Setup
 - Results
- Effectiveness
- Conclusions

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Types of Behavior-specific Knowledge (1)

- Feedback
 - An entity's (*rater*) rating of an interaction performed by a partner (*ratee*).
 - Alice: "The last service execution by Bob was very satisfying."
- Recommendation
 - An entity's (*recommender*) opinion about the previous behavior of a partner (*recommendee*)
 - Alice: "For services of type X, I can recommend Bob."

Types of Behavior-specific Knowledge (2)

- Reputation
 - General opinion of the whole network towards a single entity
→ Global characteristic of an entity.
 - Example: "With regard to services of type Y, Bob has an excellent reputation."
- Trust
 - An entity's (*truster*) degree of belief that a partner (*trustee*) will behave as expected.
 - Alice: "I trust Bob regarding services of type Z."

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Aspects of Behavior-specific Knowledge (1)

- Value $\in [-1,1]$
 - Continuous valuation allows for a finer granularity.
 - Alice:
"Bob's last service execution has been fairly good (~0.6)."
- Context
 - Allows to distinguish between different situations where entities can interact.
 - Alice: "Bob is good regarding computations of type X, but his performance wrt services of type Y has been poor."
- Facets of a context
 - Allows to distinguish between different perspectives of a context.
 - Alice: "The last service invocation has been very satisfying but also very slow."

Aspects of Behavior-specific Knowledge (2)

- Timestamp
 - Allows to emphasize the impact of current knowledge.
 - Alice: "Bob's early service executions were satisfactory but recent ones were poor."
- Certainty $\in [0, 1]$
 - Allows to quantify the certainty of an assessment.
 - Alice: "I am absolutely sure (e.g., ~ 1.0) that Bob's last performance was good."
- Estimated Effort $\in [0, 1]$
 - Allows to quantify the perceived complexity of an interaction.
 - Alice: "Bob performed simple (e.g., ~ 0.2) computations quite well but complex ones (e.g., ~ 0.9) very poorly."

Agenda

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

- Introduction
- Terminology
- Centrality
- Query Algebra for Trust
 - Idea
 - Conventional Extensions
 - Centrality Operator
- Experiments
 - Setup
 - Results
- Effectiveness
- Conclusions

Feedback Graph

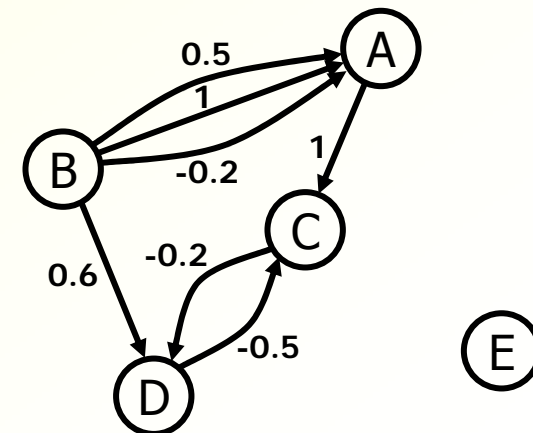
- Derivation from feedback
 - Participants → vertices
 - Feedback from A about B
→ edge from Vertex A to Vertex B
 - Value of feedback
→ weight of corresponding edge

Rater	Ratee	Value
A	C	1
B	A	0.5
B	A	1
B	A	-0.2
B	D	0.6
C	D	-0.2
D	C	-0.5



→ Characteristics of feedback graph

- Directed graph
- Not strongly connected
- Multiple edges
- Weighted edges



Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Centrality Indices (1)

- Centrality index
 - Graph-based measure to quantify the importance of a vertex according to the graph structure.
 - Different existing measures: *Indegree*, *PageRank*, *Proximity Prestige*, *HITS*, *Integration & Radiality*, etc.
 - Different measures yield different rankings.
- According to various proposals, reputation of participant = his centrality value.

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Centrality Indices (2)

Introduction

Terminology

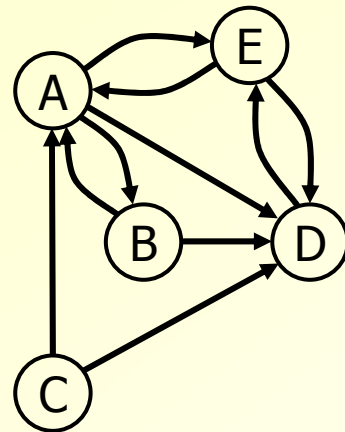
Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions



	InDegree	PageRank	Proximity
A	3	0.2	1.12
B	1	0.12	0.25
C	0	0.06	0
D	4	0.27	0.67
E	2	0.35	0.5

	InDegree	PageRank	Proximity
1	D	E	A
2	A	D	D
3	E	A	E
4	B	B	B
5	C	C	C

Centrality Measures Considered

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

- Local measures
 - consider only direct neighborhood of a vertex
 - *InDegree*
- Eigenvector-based measures
 - recursively defined measures that consider direct and indirect neighborhoods of a vertex
 - *PageRank, Authority (HITS), Positional Weakness Function*
- Distance-based measures
 - rely on shortest paths between vertices
 - *Proximity Prestige, Integration*

Requirements of CM on Input Graph

	Weighted Graphs		Unweighted Graphs (single edges)
	Multiple edges (+/- weights)	Single edges (+ weights)	
Local	✓	✓	✓
Eigenvector-based	-	✓	✓
Distance-based	-	-	✓

- **Result:**
 - Most measures require graph transformation techniques.
 - Identification of two subsequent transformation steps:
 - ♦ multiple weighted edges → single weighted edges,
 - ♦ single weighted edges → single unweighted edges.
- **Problem:** transformation incurs loss of information.

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

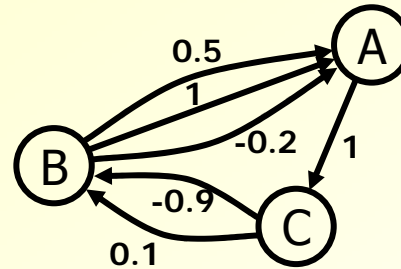
Effectiveness

Conclusions

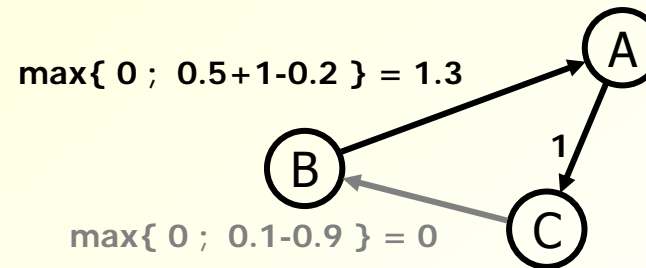
EigenTrust Transformation

- Example

- Feedback graph:



- Input graph after transformation



Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

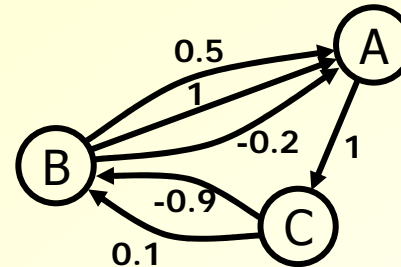
Effectiveness

Conclusions

Beta Transformation

- Example

- Feedback graph:

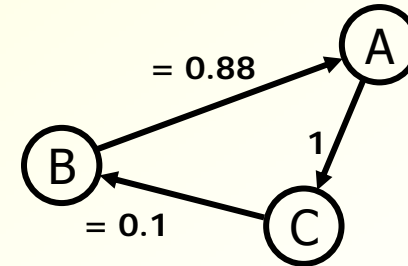


- Input graph after transformation

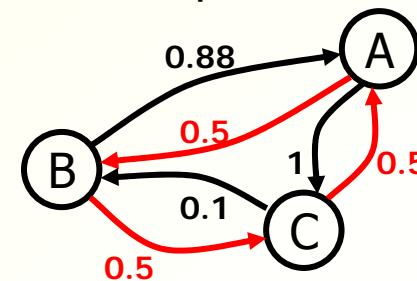
- $\beta=0$ → "no experience" = "max negative experience"

$$\frac{0.5 + 1}{|0.5| + |1| + |-0.2|}$$

$$\frac{0.1}{|0.1| + |-0.9|}$$



- $\beta=0.5$ → "no experience" = "neutral experience"



Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

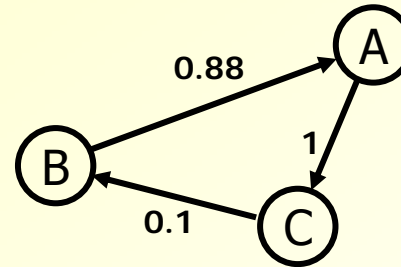
Effectiveness

Conclusions

Single weighted edges → Single Unweighted edges

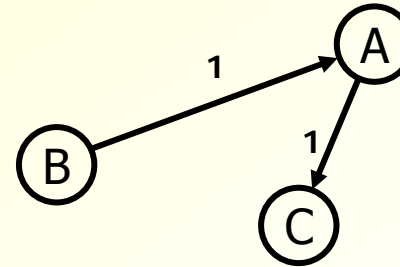
- Example

- Feedback graph:

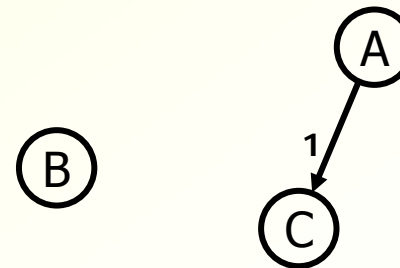


- Input graph after transformation

- $\tau=0.5$



- $\tau=0.9$



Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Impact on Trust Policy Language Envisioned

- Any centrality measure.
- Extensibility in this respect.
- Any transformation.

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Agenda

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

- Introduction
- Terminology
- Centrality
- Query Algebra for Trust
 - Idea
 - Conventional Extensions
 - Centrality Operator
- Experiments
 - Setup
 - Results
- Effectiveness
- Conclusions

Status Quo

- Existing behavior-based trust models
 - define representation of behavior-based knowledge,
 - define fixed evaluation scheme to derive the trust in a partner.
- ➔ Fixed evaluation scheme contradicts subjective nature of trust.
- Potential approach for making trust policies explicit:
Logic-based trust policy languages.
Definition of rules and clauses to derive the trustworthiness of a partner.
- ➔ Existing languages cannot satisfactorily cope with data-intensive computations required by behavior-based policies.

Introduction

Terminology

Centrality

Query Algebra
for Trust

- Idea

- Conventional
Extensions

- Centrality
Operator

Experiments

Effectiveness

Conclusions

A Framework for behavior-based trust models

- Aspects of our framework:
 - *relational representation* of behavior-specific knowledge,
 - *algebra* for the formulation of behavior-based trust policies.
- Advantages:
 - Supports definition of arbitrary user-defined trust policies for behavior-based trust models, including all existing evaluation schemes from literature we currently are aware of.
 - Relational representation allows for a straightforward implementation.

Introduction

Terminology

Centrality

Query Algebra
for Trust

- *Idea*

- Conventional
Extensions

- Centrality
Operator

Experiments

Effectiveness

Conclusions

Relational Representation of Knowledge

- Relations that represent behavior-specific knowledge: **Feedback, Recommendation, Reputation, Trust**
- Additional relation: **Entity(ID)**
- Alice: "I am quite sure that the execution of service S by Bob was good. It was a complex problem."
→ New Feedback tuple.

Rater	Ratee	Value	Context	Facet	Time	Certainty	Effort
Alice	Bob	0.95	S	Quality	12:09:45	0.75	0.8

- Goal: Trust policy language as mechanism to derive Trust, Recommendation and Reputation tuples.

Introduction

Terminology

Centrality

Query Algebra
for Trust

- Idea

- Conventional
Extensions

- Centrality
Operator

Experiments

Effectiveness

Conclusions

Relational Representation of Knowledge

- In our scenario:
 - Only Feedback tuples reflect direct experiences.
 - Other knowledge must be derived from feedback (including Trust tuples).

Introduction

Terminology

Centrality

Query Algebra
for Trust

- Idea

- Conventional
Extensions

- Centrality
Operator

Experiments

Effectiveness

Conclusions

Introduction

Terminology

Centrality

Query Algebra
for Trust

- Idea

- Conventional
Extensions

- Centrality
Operator

Experiments

Effectiveness

Conclusions

Towards an Algebra-based Policy Language

- Source: Relational representation of knowledge.
- trust policy = query on the knowledge base.
(Same with recommendation, reputation.)
- Common way to deal with relations: *Relational Algebra* (RA)
 - set of operators to be applied on relations,
 - closure property of operators allows for nesting of operators to complex algebra expressions.

➔ Basic Idea: *Relational Algebra* (RA)
as basis for our trust policy language.

Example Trust Policy

- Informal formulation:
 - "I trust individuals in context c and facet f_c if their average feedback value from the 10 most reputable entities exceeds a specific *threshold*."
 - Only feedback tuples with $\text{certainty} > 0.8$ shall be considered."

→ Algebra expression of that policy:

```
PROJECTION[trusted](
  MAP[trusted, (avg_value > threshold)](
    GROUP[avg_value, AVG(Feedback.value), {ratee}](
      JOIN[Feedback.rater=Reputation.entity](
        TOP[10, Reputation.value](
          SELECTION[context=c, facet=f_c](Reputation)
          SELECTION[ratee=idpartner context=c, facet=f_c, certainty>0.8](Feedback)
        ) ) );
```

Introduction

Terminology

Centrality

Query Algebra
for Trust

- Idea

- Conventional
Extensions

- Centrality
Operator

Experiments

Effectiveness

Conclusions

Algebra-based Policy Language

- Observation:
 - Basic operators of the RA are not sufficient to formulate behavior-based trust policies.
 - Extension with additional operators are necessary.
- ➔ Which further operators are essential to arrive at expressiveness desired?
- First step: Existing additional operators from literature
 - Top operator (e.g., Bertino et al., 2004)
 - Map operator (e.g., Aberer and Fischer, 1995)

Introduction

Terminology

Centrality

Query Algebra
for Trust

- Idea

- Conventional
Extensions

- Centrality
Operator

Experiments

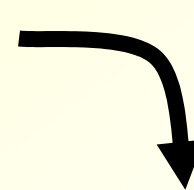
Effectiveness

Conclusions

Conventional Extensions to the RA (1)

- Top Operator: **TOP**[k,attr](relation)
 - returns the k tuples with the highest value of a attribute $attr$
 - Example:

ID	...	Value
Bob	...	0.71
Carol	...	0.95
Alice	...	0.98
Eve	...	0.75
Dave	...	0.90



TOP[3, Value](Reputation)

ID	...	Value
Carol	...	0.95
Alice	...	0.98
Dave	...	0.90

Introduction

Terminology

Centrality

Query Algebra
for Trust

- Idea

- Conventional
Extensions

- Centrality
Operator

Experiments

Effectiveness

Conclusions

Conventional Extensions to the RA (2)

- Map Operator: **MAP**[attr,expression(A_1, \dots, A_n)](relation)
 - Allows the execution of user-defined functions over the attributes of a relation.
 - The functions are applied separately to each tuple; the results become a new attribute.
 - Example:

Rater	Ratee	...	Value	Effort
Alice	Bob	...	1.0	0.2
Alice	Carol	...	0.8	0.9

MAP[Weighted, (Value*Effort)](Feedback)

Rater	Ratee	...	Value	Effort	Weighted
Alice	Bob	...	1.0	0.2	0.2
Alice	Carol	...	0.8	0.9	0.72

Introduction

Terminology

Centrality

Query Algebra
for Trust

- Idea

- Conventional
Extensions

- Centrality
Operator

Experiments

Effectiveness

Conclusions

An Operator for Centrality Computation

- Requirements for a centrality operator:
 - Flexible specification of the underlying graph

Rater	Ratee	...	Value	Effort	Weighted
Alice	Bob	...	1.0	0.2	0.2
Alice	Carol	...	0.8	0.9	0.72

e.g., choice of the weight of an edge: "Value" vs. "Weighted"

- Support of various centrality measures within one operator

➔ Definition of centrality operator:

CENTRALITY[attr, A_v , A_s , A_t , A_w , Measure](R_{vertices} , R_{edges})

new attrib

vertex

source

target

weight

Introduction

Terminology

Centrality

Query Algebra
for Trust

- Idea

- Conventional
Extensions

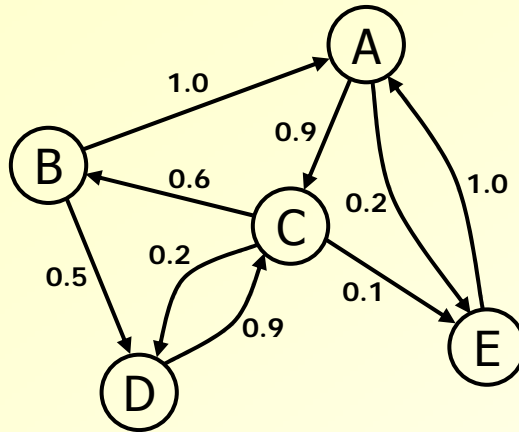
- Centrality
Operator

Experiments

Effectiveness

Conclusions

Centrality Operator - Example



Recommendation

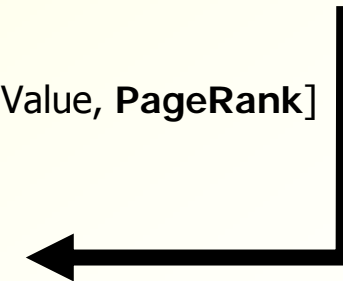
Recommender	Recommendee	...	Value
A	C	...	0.9
A	E	...	0.2
B	A	...	1.0
B	D	...	0.5
C	B	...	0.6

Entity

ID
A
B
C
D
E

CENTRALITY[PageRank, ID, Recommender, Recommendee, Value, PageRank]
(Entity, Recommendation)

ID	PageRank
A	0.23
B	0.21
C	0.31
D	0.15
E	0.1



- Introduction
- Terminology
- Centrality
- Query Algebra for Trust
- Idea
- Conventional Extensions
- Centrality Operator
- Experiments
- Effectiveness
- Conclusions

Agenda

- Introduction
- Terminology
- Centrality
- Query Algebra for Trust
 - Idea
 - Conventional Extensions
 - Centrality Operator
- Experiments
 - Setup
 - Results
- Effectiveness
- Conclusions

Centrality Operator

- Nature of centrality computation
 - Very time-consuming and resource-intensive.
 - ➔ Centrality computation is the most costly part of evaluation of a trust policy.
- Implemented centrality measures in PL/SQL (Oracle 10g)
 - *PageRank, Positional Power Function* (eigenvector centrality measures based on power iteration implementation)
 - *Authorities, Proximity Prestige, Integration*
- Experiments
 - Efficiency: Performance of our implementations
 - Quality of Centrality Measures: Comparison of ranking results

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

- Setup

- Results

Effectiveness

Conclusions

Creation of a Reference Ranking

- Core component: *feedback generator*
 - Simulates interactions between participants
 - Behavior of a participant specified by its cooperation value (= probability that participant cooperates)
 - Generation of feedback based on outcome of interactions (outcome depends on cooperation values)
- ➔ Participants sorted by coop values: *Reference ranking*
- ➔ Computation of CM on feedback: *Result rankings*
- Intuition: "Good" measures assign high ranks to participants with high cooperation value (and vice versa)

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

- Setup

- Results

Effectiveness

Conclusions

Quantifying the Distance between 2 Rankings

- Participants in reputation systems use cutoff strategies, i.e., a partner B is...
 - trustworthy, if rank of B is above a threshold k (in Top- k list),
 - untrustworthy, if rank of B is below k (not in Top- k list)

(exact rank of B is unimportant)

→ A participant can make 2 wrong decisions

- Deeming an untrustworthy partner trustworthy
- Deeming an trustworthy partner untrustworthy

Our measure: fraction of wrong decisions (FWD).

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

- Setup

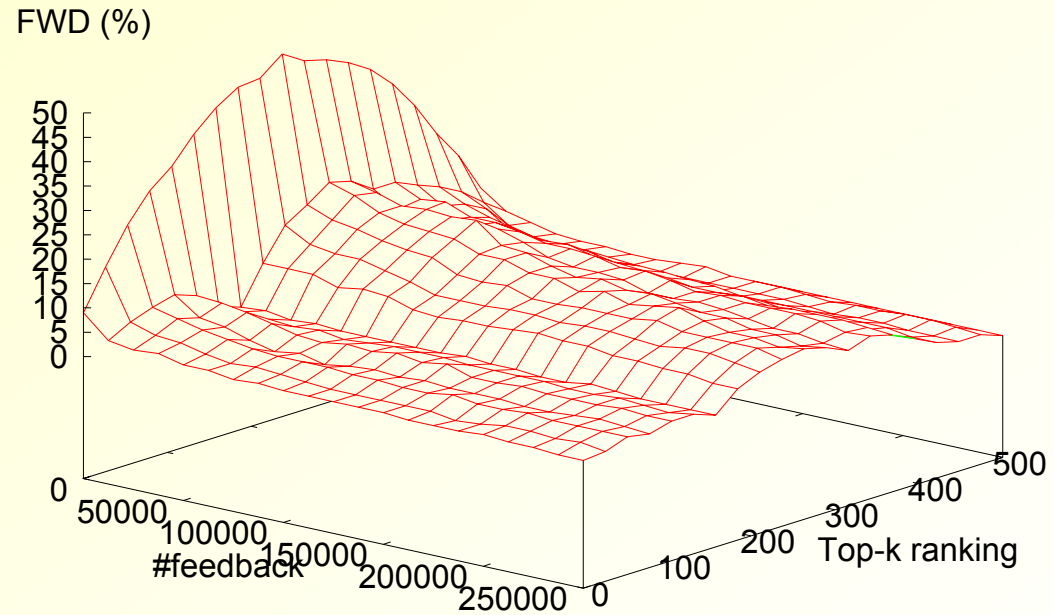
- Results

Effectiveness

Conclusions

Amount of Feedback Data Needed

- Example:
PageRank



- Results
 - A lot of feedback required to gain no more improvement
 - Quite good results with 25k feedback items (50 per peer)
 - ➔ Rather "slow" if participants change their behavior frequently

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

- Setup

- Results

Effectiveness

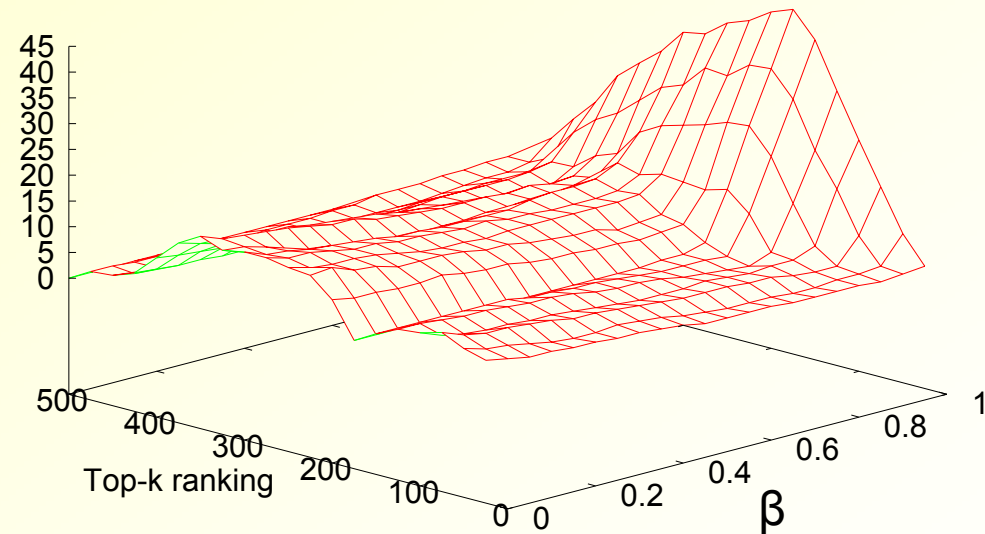
Conclusions

Dealing With "No Experience"

- Parameter β

- Example:
PageRank

FWD (%)



- Results:

- "Neutral knowledge" ($\beta=0.5$) yields best results
- $\beta > 0.85$ (deeming unknown participants good) is inappropriate
- $\beta=0$ yields quite good result + allows pruning of edges
→ Trade-off: Quality vs. computation performance

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

- Setup

- Results

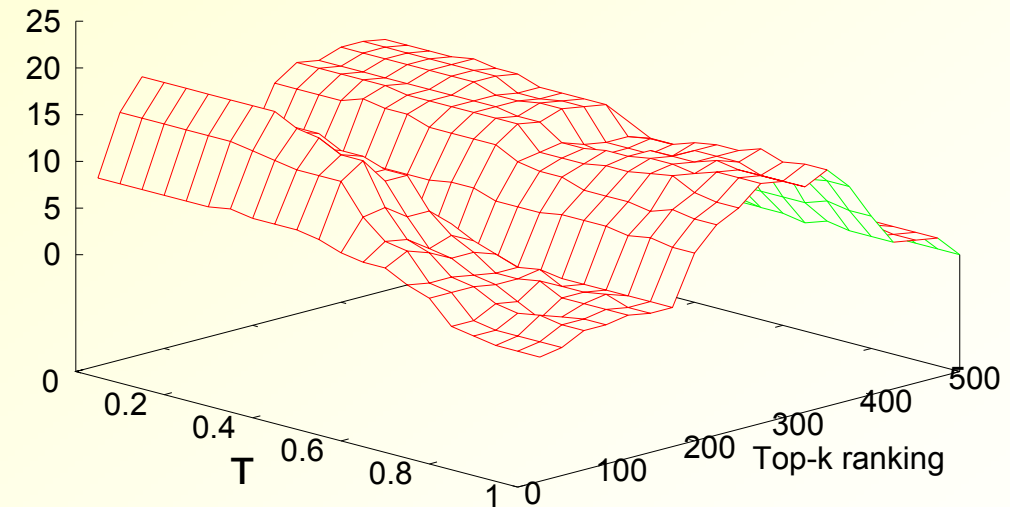
Effectiveness

Conclusions

Pruning Weighted Edges

- Parameter τ FWD (%)

- Example:
Proximity



- Results

- $\tau > 0.5$: significant improvement for small Top- k lists (explanation: more importance on "strong" edges)
- The larger τ the more edges can be pruned

➔ "Good" values of τ can improve quality + performance

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

- Setup

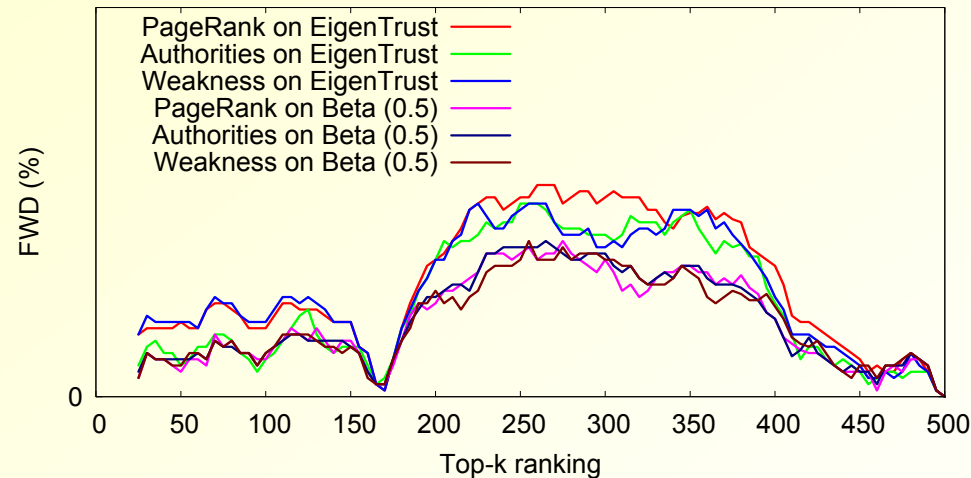
- Results

Effectiveness

Conclusions

EigenTrust- vs. Beta-Transformation

- Eigenvector-based measures



- Results:

- In general, Beta-transformation is superior
- Quality of EigenTrust-transformation depends on ratio of uncooperative participants (no distinction between "no experience" and "bad experience")

- Different degrees concerning loss of knowledge verifiable

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

- Setup

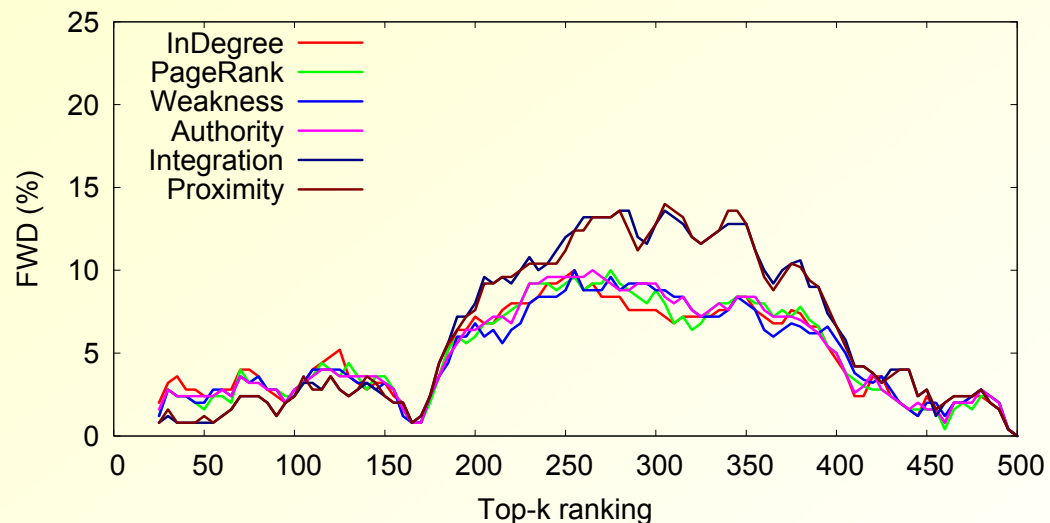
- Results

Effectiveness

Conclusions

Comparison of All Centrality Measures

- $\beta=0.5$
 $\tau=0.95$



- **Results**

- All measures yield similar results, esp. for smaller Top-k lists
- Distance-based measures profit from large value of τ
- ➔ With good β, τ : All Measures quite suitable
- Without graph:
 - ♦ Performances differ significantly
 - ♦ Distance-based measures perform poorest
- ➔ Application of distance-based measures in larger networks questionable.

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

- Setup

- **Results**

Effectiveness

Conclusions

Insights from Experiments

- No centrality measure is clearly better.
- Both effectiveness and efficiency (in terms of computation effort) are important.
- Much feedback needed to have low FWD.
 - Policies need to be sophisticated.
 - Trust policy language must facilitate this.

Agenda

- Introduction
- Terminology
- Centrality
- Query Algebra for Trust
 - Idea
 - Conventional Extensions
 - Centrality Operator
- Experiments
 - Setup
 - Results
- Effectiveness
- Conclusions

Effectiveness

- Which combinations of policies are successful and lead to stable and efficient (in the economic sense) systems?
- Which policies do humans actually use in different situations?

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Effectiveness – Experiments (1)

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

- Human participants playing a game:
 - System-controlled entities interact with each other.
 - An entity may issue one service request per round.
 - Entity processing a service request is chosen randomly.
 - Each player may specify trust policy of 'his' entity.
- Costs/benefits:
 - processing a service request incurs costs,
 - issuing a service request and issuing feedback are free,
 - having a service invocation processed yields benefit,
 - payment based on performance.

Effectiveness – Experiments (2)

- Humans enter 'their' trust policy in natural language.
- We translate these statements to algebra expressions.
- Different experiments, with different information available.

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Firefox


http://i40lx08.ipd.uni-karlsruhe.de/CoSim/index.php?Show=UpdatePolicy

Greyhoun... Essex Inn ... Arlington ... Online Re... Car Renta... CERIAS - ... LEO D-E E... The S...

The Service Game. How Good is Your Strategy?

Aktuell	Spielanleitung	Wissensbasis	Ergebnisse	Policies	Abmelden
---------	----------------	--------------	------------	----------	----------

Formulierung einer neuen Policy




Es wird gerade gespielt. Daher können Sie aktuell die Policy für Ihren Stellvertreter-Rechner nicht ändern. Sie können allerdings weiter neue Policies formulieren und alte, bis auf die aktuell aktive Policy, entfernen.

Im folgenden Textfeld können Sie ein neue Policy in natürlicher Sprache formulieren.

[Neue Policy übernehmen](#)

Hier Ihre bereits formulierten Policies

Ich vertraue allen, außer sie haben meine Aufträge abgelehnt. **AKTIV**

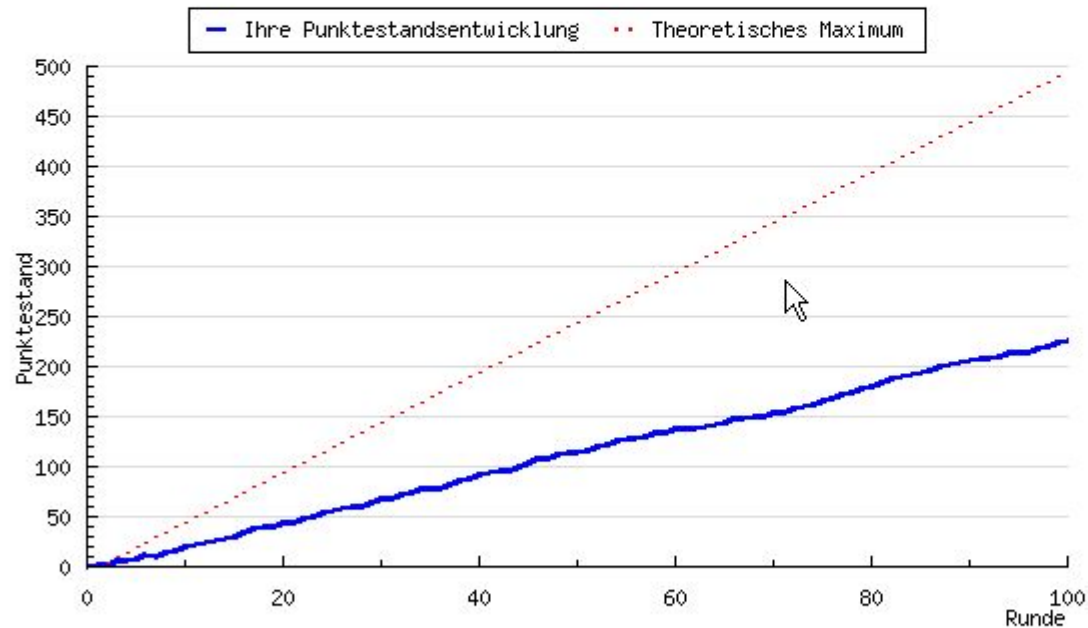


Olymp

The Service Game developed by Christian von der Weth, Thorben Burghardt, and Prof. Klemens Böhm, Universität Karlsruhe (TH)
Contact: Christian von der Weth (weth AT ipd DOT uni MINUS karlsruhe DOT de)

Punktstandsentwicklung

In untenstehender Graphik sehen Sie die Entwicklung Ihres eigenen Punktstandes (blau) sowie das theoretische Maximum (rot), was Sie hätten erreichen können (das theoretische Maximum wird erreicht, wenn Sie alle Ihre Aufträge bearbeitet bekommen, selbst aber keine Aufträge bearbeiten).



Kreta

Insights So Far

- Same policies may result in different rankings in different runs of the game.
- Players have not yet resorted to centrality measures to formulate policies – so far.
- The more information on others is available, the more 'hectic' players become.

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Agenda

- Introduction
- Terminology
- Centrality
- Query Algebra for Trust
 - Idea
 - Conventional Extensions
 - Centrality Operator
- Experiments
 - Setup
 - Results
- Effectiveness
- Conclusions

Summary

- The following questions are fundamental:
 - How to establish trust in distributed systems?
 - How do individuals (humans) interact in such settings?
 - How should a language look like that allows to formalize these issues?
 - How to evaluate expressions in the language efficiently?

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Summary

- What have we done so far?
 - Collected various meaningful behavior-based trust policies from literature and our own attempts.
 - Motivation of an algebra-based approach for formulation of behavior-based trust policies.
 - Definition of a relational representation of behavior-specific knowledge.
 - Definition of a query algebra for trust
 - ◆ Listing of necessary operators from literature (basic operators from the RA incl. existing extensions),
 - ◆ Definition of a centrality operator to compute various centrality measures.
 - Presentation of experimental results.
 - Design of setting for effectiveness experiments.

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions

Outlook

- Declarative language instead of algebra.
- Efficiency issues, in particular query optimization (equivalences of algebra expressions, selectivity estimation, view materialization, multi-query optimization).
- Distribution.
- Privacy.
Relationship between privacy and economic success.

Introduction

Terminology

Centrality

Query Algebra
for Trust

Experiments

Effectiveness

Conclusions