Effect of Heuristics on Serendipity in Path-Based Storytelling with Linked Data

Laurens De Vocht[1(✉)], Christian Beecks[2], Ruben Verborgh[1], Erik Mannens[1], Thomas Seidl[2], and Rik Van de Walle[1]

[1] Department of Electronics and Information Systems, Ghent University - iMinds, Sint-Pietersnieuwstraat 41 B1, 9050 Ghent, Belgium
{laurens.devocht,ruben.verborgh,erik.mannens,rik.vandewalle}@ugent.be
[2] Data Management and Data Exploration Group, RWTH Aachen University, Department of Computer Science 9, 52056 Aachen, Germany
{beecks,seidl}@cs.rwth-aachen.de

Abstract. Path-based storytelling with Linked Data on the Web provides users the ability to discover concepts in an entertaining and educational way. Given a query context, many state-of-the-art pathfinding approaches aim at telling a story that coincides with the user's expectations by investigating paths over Linked Data on the Web. By taking into account serendipity in storytelling, we aim at improving and tailoring existing approaches towards better fitting user expectations so that users are able to discover interesting knowledge without feeling unsure or even lost in the story facts. To this end, we propose to optimize the link estimation between - and the selection of facts in a story by increasing the consistency and relevancy of links between facts through additional domain delineation and refinement steps. In order to address multiple aspects of serendipity, we propose and investigate combinations of weights and heuristics in paths forming the essential building blocks for each story. Our experimental findings with stories based on DBpedia indicate the improvements when applying the optimized algorithm.

Keywords: Storytelling · Serendipity · Pathfinding · A* · Linked data · Heuristics

1 Introduction

Algorithmic storytelling can be seen as a particular kind of querying data. Given a set of keywords or entities, which are typically, but not necessarily dissimilar, it aims at generating a story by explicitly relating the query context with a path that includes semantically related resources. Storytelling is utilized for example in entertaining applications and visualizations [21] in order to enrich related Linked Data resources with data from multimedia archives and social media [9] as well as in scientific research fields such as bio-informatics where biologists try to relate sets of genes arising from different experiments by investigating the implicated pathways [16] or discovering stories through linked books [7].

The aspects that make a story a good story are captured in the term *serendipity*. The term depicts a mixture between casual, lucky, helpful and unforeseen facts, also in an information context [11]. In fact, users want to be surprised and they want to discover, confirm, and extend knowledge - but not feel unsure while doing so. This means that users can always relate presented story facts to their background knowledge.

In order to generate a story, graph-based pathfinding approaches are typically utilized. The most frequently encountered algorithm to determine a path between multiple resources is the A* algorithm [14]. This algorithm, which is based on a graph representation of the underlying data (i.e., resources and links between them define nodes and edges, respectively) determines an optimal solution in form of a lowest-cost traversable path between two resources. The optimality of a path, which is guaranteed by the A* algorithm, does not necessarily comply with the users' expectations.

By considering for instance large real-world semantic graphs, such as Linked Data graphs, where links between nodes are semantically annotated, users are able to directly interpret the transitions between nodes and thus the meaning of a path. Caused by the inevitable increasing number of nodes and sometimes loosely related links among them, optimal paths frequently show a high extent of *arbitrariness*: paths appear to be determined by chance and not by reason or principle and are often affected by resources that share many links. In addition, large real-world semantic graphs typically exhibit small-world properties. Applying pathfinding approaches increases arbitrariness due to the large number of possible relations that connect two entities in a query context.

In order optimize the serendipity level of the storytelling and to mitigate arbitrariness of a story, we propose an in-depth extension of our algorithm [8], embedded in the Everything is Connected Engine (EiCE) [9]. In fact, our contribution is twofold: (i) We outline the extended algorithm which reduces arbitrariness by increasing the relevance of links between nodes through additional pre-selection and refinement steps; and (ii) we discuss the reorganization of code execution between client and server utilizing Linked Data Fragments. We conclude our paper with preliminary results and an outlook on future work.

2 Related Work

We divide related work into two categories: (i) retrieving associations, and (ii) ranking associations. The former category considers approaches for retrieving semantic associations, with a particular focus on paths, while the latter category considers methods to rank semantic associations.

Retrieving Associations. The original implementation[1] of the "Everything is Connected Engine" (EiCE) [9] uses a distance metric based on the Jaccard for

[1] http://demo.everythingisconnected.be/.

pathfinding. It applies the measure to estimate the similarity between two nodes and to assign a random-walk based weight, which ranks more rare resources higher, thereby guaranteeing that paths between resources prefer specific relations over general ones [18]. The A* algorithm is applied for revealing relations between Linked Data resources to recombine data from multimedia archives and social media for storytelling. In contrast to the EiCE system, which heuristically optimizes the choice of relationship explanations, the REX system [10] identifies a ranked list of relationship explanations. A slightly different approach with the same goal of exploratory association search is Explass [5]. It provides a flat list (top-K) clusters and facet values for refocusing and refining a search. The approach detects clusters by running pattern matches on the datasets to compute frequent, informative and small overlapping patterns [5]. All of these approaches where investigated on DBpedia.

Ranking Associations. The number of possible combinations to fill in the bridging resources between entities in the a knowledge base such as DBpedia is much larger than the number of entities themselves. Thus, the likelihood that this type of queries would result in an overwhelming number of possible results for users is increased. Furthermore, it is unlikely that traditional ranking schemes for ranking results may be applied to a graph representation [4]. Ranking semantic associations is different from ranking documents. In general, document ranking according to relevance focuses on the match degree of search keywords (without formal semantics). When ranking semantic associations, approaches semantically reinterpret query results in relation to the query context by using semantic distance (or similarity) to the datasets or search graph. Alternatively, a ranking can vary from rare relationships discovery mode to common relationships in conventional mode [3]. Techniques that support context driven ranking take into account ontological relations of the result instances in respect to the query context [2].

3 Pathfinding for Storytelling

Each path that contributes to a story is determined within a query context comprising both start and destination resources. Our algorithm reduces the arbitrariness of a path between these resources by increasing the *relevance* of the links between the nodes using a domain-delineation step. The path is refined by iteratively applying the A* algorithm and with each iteration attempting to improve the overall semantic relatedness between the resources until a fixed number of iterations or a certain similarity threshold is reached.

3.1 Domain Delineation

Instead of directly initializing the graph as-is by including all links between the resources, we identify the relevance of predicates with respect to the query context. This is done by extracting and giving higher preference to the type of relations (predicates) that occur frequently in the query context. In this way, we make sure that the links included in the story matter because each predicate

```
Data: start, destination, graph, k
Result: list of important predicates given the context
initialize pf_irf_p_list;
predicates_start = unique predicates start;
predicates_dest = unique predicates destination;
predicates_considered = intersection predicates_start predicates_dest;
foreach predicates_considered as p do
    pf_irf_p = compute pf_irf p;
    add pf_irf_p to list
end
reverse sort pf_irf_p_list;
take the first k elements of the list as important predicated;
```

Algorithm 1. Important predicate selection

that describes the semantics of a link also occurs in the direct neighborhood of the query context. The selection of the most important predicates for domain delineation is shown in Algorithm 1.

In order to select the links in a graph that are most relevant based on the given start and destination nodes, we utilize an adapted variant of the TF/IDF [1] measure: PF/IRF. The PF/IRF measure reflects the importance of a predicate with respect to a resource in a dataset and is defined as follows:

$$PF(p) = \frac{\text{Number of times predicate } \mathbf{p} \text{ appears in a resource}}{\text{Total number of predicates linked to the resource}} \tag{1}$$

$$IRF(p) = \ln \frac{\text{Total number of resources}}{\text{Number of resources with predicate } \mathbf{p} \text{ in it}} \tag{2}$$

For example, the PF/IRF computation for predicates linked to *Carl Linnaeus* is explained below for the case when PF/IRF is determined in the context of start *Carl Linnaeus* and destination *Charles Darwin* based on DBpedia.

1. We determine predicates that are important in the context. This is done by retrieving the distinct predicates that are linked to the context nodes.
2. For each predicate, we compute its occurrence based on linked nodes. In addition, the total number of predicates linked to the resource *Carl Linnaeus* is determined.
3. As a result, the total number of predicates linked to the resource *Carl Linnaeus* is 9890. For the predicates *binomialAuthority* and *label* we obtain the values 2297 and 12, respectively. The total number of resources (including objects) in the DBpedia is $M = 27,318,782$.
4. We compute the number of resources which are linked using each predicate by counting the distinct number of resources through the predicate *binomialAuthority* and *label* in both directions. This results in $155,207$ and $10,471,330$ respectively.
5. By using the PF/IRF formula above we finally get the following values: PF/IRF(*binomialAuthority*) $= 2297/9890 * \ln(27,318,782/155,207) = \mathbf{1.20}$ and PF/IRF(*label*) $= 12/9890 * \ln(27,318,782/10,471,330) = \mathbf{0.0011}$

Since the PF/IRF value of *binomialAuthority* is much higher than that of *label*, the predicate *binomialAuthority* is more likely to be included.

3.2 Algorithm

The output of the aforementioned domain delineation step can be thought of a Linked Data graph comprising nodes and predicates which are semantically related to the user's query context. In order to provide a serendipitous story based on this Linked Data graph, the graph has to be traversed via a meaningful path including the start and end destination of the query context. A single or multiple paths are then used as essential building blocks for generating a story.

In order to find a path in a Linked Data graph, we utilize the A* algorithm due to its ability of computing an optimal solution, i.e., a (shortest) cost-minimal path between two nodes with respect to the weights of the linking predicates contained in the path. In order to reduce the number of predicates to be examined when computing the lowest-cost path between two nodes and, thus, to achieve an improvement in the computation time of the A* algorithm, heuristics are frequently used to determine the order of expansion of the nodes according to the start and end node provided within the query context. In addition to a heuristic, the A* algorithm utilizes a weighting function in order to determine paths which are semantically related to source and destination nodes as specified within the query context. Thus, the serendipity of a story generated based on a single or multiple paths is strongly connected to the underlying weighting scheme and heuristic. In the following section, we propose and investigate various heuristics before we will introduce different weighting schemes.

3.3 Heuristics

The objective of a heuristic is to determine whether a node in a Linked Data graph is semantically related to the query context, i.e. source and destination nodes, and thus a good choice for expansion within the A* algorithm. For this purpose, we formally define a heuristic as a function $heuristic : G \times G \rightarrow \mathbb{R}$ that assigns all pairs of nodes $n_a, n_b \in G$ from a Linked Data graph G a real-valued number indicating their semantic relation.

Jaccard Distance. The first heuristic we consider is the **Jaccard distance** which is a simple statistical approach taking into account the relative number of common predicates of two nodes. The higher the number of common predicates, the more likely similar properties of the nodes and thus the semantically closer in terms of distance the corresponding nodes. The Jaccard distance $jaccard : G \times G \rightarrow \mathbb{R}$ is defined for all nodes $n_a, n_b \in G$ as follows:

$$jaccard(n_a, n_b) = 1 - \frac{\|n_a \cap n_b\|}{\|n_a \cup n_b\|} \tag{3}$$

Normalized DBpedia Distance. Another approach that can be utilized as a heuristic is the **Normalized DBpedia Distance** [13,19]. This approach adapts the idea of the Normalized Web Distance to DBpedia and considers two nodes n_a and n_b to be semantically similar if they share a high number of common neighboring nodes linking to both n_a and n_b. The Normalized DBpedia Distance $NDD : G \times G \rightarrow \mathbb{R}$ is defined for all nodes $n_a, n_b \in G$ as

$$NDD(n_a, n_b) = \frac{\max(\log f(n_a), \log f(n_b)) - \log f(n_a, n_b)}{\log N - \min(\log f(n_a), f(n_b))}, \tag{4}$$

where $f(n) \in \mathbb{N}$ denotes the number of DBpedia nodes linking to node $n \in G$, $f(n, m) \in \mathbb{N}$ denotes the number of DBpedia nodes linking to both nodes n and $m \in G$, and where the constant N is defined as the total number of nodes in DBpedia, which is about $2.5M$.

Confidence. Another heuristic that has been proposed for semantic path search in Wikipedia is the **Confidence measure** [12]. The Confidence measure is an asymmetrical statistical measure that can be thought of as the probability that node n_a occurs provided that node n_b has already occurred. The Confidence measure $P : G \times G \rightarrow \mathbb{R}$ is defined for all nodes $n_a, n_b \in G$ as:

$$P(n_a | n_b) = \frac{f(n_a, n_b)}{f(n_b)} \tag{5}$$

As opposed to the heuristics, which affect the expansion order within the A* algorithm by estimating the potential semantic relatedness of a node, weighting schemes are finally utilized in order to asses the quality of a path. We propose different weighting schemes in the following section.

3.4 Weights

The objective of a weighting function is to determine the exact cost of a path, which is the sum of weights of linking nodes. A weighting is formalized as a function $weight : G \times G \rightarrow \mathbb{R}$ between the corresponding nodes from the Linked Data graph.

Jaccard Distance. We apply the **Jaccard distance** in exactly the same way to determine the weights so that the core algorithm prefers similarity in adjacent nodes in each path. We use this distance between two directly adjacent nodes rather than unconnected nodes in the graph.

Combined Node Degree. Moore et al. [18] proposed the **combined node degree** which can be used to compute a weight that encourages rarity of items in a path. It ranks more rare resources higher, thereby guaranteeing that paths between resources prefer specific relations. The main idea is to avoid that paths go via generic nodes. It makes use of the node degree, the number of in and outgoing links. The combined node degree $w : G \times G \rightarrow \mathbb{R}$ is defined for all nodes $n_a, n_b \in G$ as:

$$w(n_a, n_b) = \log(\deg(n_a)) + \log(\deg(n_b)) \tag{6}$$

Jiang and Conrath Distance. Mazuel and Sabouret [17] suggest to take into account the object property ontology relation between two adjacent items in a path. The base distance measure there is the **Jiang and Conrath distance** [15], which we can interpret in terms of RDF by looking at the classes of each of the nodes and determining the most common denominator of those classes in the ontology. Once this type is determined, the number of subjects that exist with this type is divided by the total number of subjects. The higher this number, the more generic the class, thus the more different two nodes.

3.5 Refinement

After a path is determined by the A* algorithm, we measure the semantic relatedness, corresponding to the lowest semantic distance between all resources occurring in the path with respect to the query context. This done for example by counting the number of overlapping predicates (i) among each other combined with those in the start and destination resources; and then (ii) averaging and normalizing this count over all resources. Depending on the threshold and the maximum number of iterations configured, this process is repeated, typically between 3 and 10 times. Finally, the path with the shortest total *distance* (or cost) is selected for the story. The *distance* for a $path = (s_1, s_2, ..., s_n)$ is computed based on a weight function w as $\text{distance}(path) = \frac{\sum_1^{n-1} w(s_i, s_{i+1})}{n}$.

4 Implementation and Presentation of Stories

The complexity of this approach is enforced by is the centrality of underlying graph-indexing and data-processing algorithms. It turns out that server-side query processing degrades the performance of a server and therefore limits its *scalability*. While many approaches are suitable for a small-to-moderate number of clients, they reveal to be a performance bottleneck when the number of clients is increased.

Instead of running the algorithm entirely on the server, we moved CPU and memory intensive tasks to the client. The server translates user queries into smaller, digestible fragments for the data endpoint. All optimizations and the execution of the algorithm are moved to the client. This has two benefits: (i) the CPU and memory bottleneck at server side is reduced; and (ii) the more complex data fragments to be translated stay on the server even though they do not require much CPU and memory resources, but they would introduce to many client-side requests.

A separate index with linked data documents to store the fragments for fast navigating graphs served a first iteration but turned out to be only limited scalable. It required each time a pre-selection of datasets that would need to be manually or semi automatically scheduled to be ingested or updated. The improved algorithm[2] runs using Triple Pattern Fragments (TPFs)[22]. TPF pro-

[2] The original algorithm can be found at https://github.com/mmlab/eice and the improved algorithm at https://www.npmjs.com/package/everything_is_connected_engine.

vides a computationally inexpensive server-side interface that does not overload the server and guarantees high availability and instant responses. Basic triple patterns (i.e. *?s ?p ?o*) suffice to navigate across linked data graphs (no complex queries needed).

Obviously a set of paths is not a presentable story yet. We note that even if a path comprise just the start and destination (indicating they are linked via common hops or directly to each other), the story will contain interesting facts. This is because each step in the path is separated with at least one hop from the next node. For example, to present a story about *Carl Linnaeus* and *Charles Darwin*, the story could start from a path that goes via *J.W. von Goethe*. The resulting statements serve as basic facts, which are relation-object statements, that make up the story. It is up to the application or visualization engine to present it to end-users and enrich it with descriptions, media or further facts. Table 1 exemplary explicates the idea of statements as story facts.

Table 1. The statements as story facts

About	Relation	Object
Carl Linnaeus and Charles Darwin	are	scientists
J.W. von Goethe	influenced	Carl Linnaeus and Charles Darwin
J.W. von Goethe and Charles Darwin	influenced	Karl Marx and Sigmund Freud

5 Evaluation

To determine whether the arbitrariness of a story is reduced, we validated that our optimization improved the link estimation between concepts mentioned in a story. To this end, we computed stories about the four highest ranked DBpedia scientists, according to their PageRank score[3]. Resources with a high PageRank are typically very well connected and have a high probability to lead to many arbitrary paths.

5.1 Initial Sample

We have determined the pairwise semantic relatedness of the story about them by applying the Normalized Google Distance (NGD). The results are shown in Table 2.

Table 2 shows that the entities *Aristotle* and *Physics* are included in the story when applying the original algorithm. These entities are perfect examples of *arbitrary* resources in a story which decreases the consistency. Except that they are related to science, it is unclear to the user why the algorithm ‘reasoned’ them to be in the story. When utilizing the optimized algorithm these entities are replaced by *J._W._Von_Goethe* and *D._Hume*.

[3] http://people.aifb.kit.edu/ath#DBpedia_PageRank.

Table 2. The comparison between the original and optimized algorithm shows that the semantic relatedness can be improved in all cases except for the last two when the entities were already closely related, their NGD in the original algorithm was already relatively low.

No.	Query Context	Original Algorithm	NGD	Optimized Algorithm	NGD
S1	C._Linnaeus - C._Darwin	C._H._Merriam	0.50	J._W._Von_Goethe	**0.43**
S2	C._Linnaeus - A._Einstein	Aristotle	0.70	J._W._Von_Goethe	**0.45**
S3	C._Linnaeus - I._Newton	P._L._Maupertuis	0.48	D._Diderot	**0.40**
S4	A._Einstein - I._Newton	Physics	0.62	D._Hume	**0.45**
S5	C._Darwin - I._Newton	D._Hume	**0.38**	Royal_Liberty_School	0.40
S6	C._Darwin - A._Einstein	D._Hume	**0.43**	B._Spinoza	0.44

5.2 Detailed Sample

In order to verify our results, we also include the total semantic similarity of a path by computing the semantic relatedness between all neighboring node pairs in that path. As can be seen in Table 2, the optimized algorithm seemed to be able to improve the link estimation of the resulting paths. To evaluate the results we used three different similarity measures: W2V[4], NGD [6], and SemRank [3,4].

We used an online available Wiki2VecCorpus using vectors with dimension 1000, no stemming and 10skipgrams[5]. We computed the similarities based on that model by using *gensim*[6]. We implemented the NGD - generalized as the normalized web search distance, on top of the Bing Search API, using the same formula as depicted in the heuristic for the algorithm.

We applied SemRank to evaluate the paths, in particular to capture the serendipity of each path. The serendipity is measured by using a factor μ to indicate the so called 'refraction' how different each new step in a path is compared to the previous averaged over the entire path. Furthermore the information gain is modulated using the same factor μ. The information gain is computed from the weakest point along the path and an average of the rest. So that we get as formula for SemRank and a path p:

$$\text{SemRank}(\mu, p) = [\frac{1-\mu}{I(p)} + \mu I(p)] \times [1 + \mu R(p)], \tag{7}$$

where $I(p)$ is the overall information gain in the path and $R(p)$ is the average refraction. There are three special cases [4]: (i) **conventional** with $\mu = 0$ leading to $\text{SemRank}(0, p) = \frac{1}{I(p)}$, serendipity plays no role and so no emphasis is put one newly gained or unexpected information; (ii) **mixed** with $\mu = 0.5$ leading to $\text{SemRank}(0.5, p) = [\frac{1}{2I(p)} + \frac{I(p)}{2}] \times [1 + \frac{R(p)}{2}]$, a balance between unexpected and newly gained information; and (iii) **discovery** with $\mu = 1$ leading to

[4] https://code.google.com/p/word2vec/.
[5] https://github.com/idio/wiki2vec.
[6] https://radimrehurek.com/gensim/.

Table 3. Abbreviations explained and short interpretation of the measures used.

Abbreviation	Description
W2Vs	Word2Vector similarity using Wikipedia English Corpus
NGD	Normalized Web Search Distance using Bing API
SR-C	SemRank - Conventional - No particular role for serendipity
SR-M	SemRank - Mixed - Serendipity plays partly a role
SR-D	SemRank - Discovery - Serendipity has a major role
PR	PageRank - Centrality Degree of a Node

Table 4. Detailed comparison between the original and optimized algorithm.

	Measure	Higher Better?	S1	S2	S3	S4	S5	S6	AVG	STDEV
Original	SR-C	+	6.46	6.70	5.48	9.47	6.50	9.00	7.17	1.59
	SR-M	+	4.04	4.05	3.34	5.25	4.11	5.21	4.35	0.75
	SR-D	+	0.22	0.20	0.25	0.13	0.23	0.14	0.20	0.05
	NGD	–	0.64	0.69	0.48	0.31	0.48	0.29	0.48	0.16
	W2Vs	+	?	?	0.18	0.32	0.21	0.39	0.20	0.02
	PR	–	*2631.89*	66.27	179.50	62.39	357.36	62.39	166.38	128.58
Improved	SR-C	+	9.19	8.00	7.17	6.74	9.47	6.50	7.78	1.15
	SR-M	+	5.39	4.70	4.00	3.98	5.44	3.95	4.52	0.65
	SR-D	+	0.14	0.16	0.17	0.19	0.13	0.21	0.17	0.03
	NGD	–	0.53	0.22	0.60	0.38	0.32	0.55	0.45	0.14
	W2Vs	+	0.21	0.19	0.20	?	0.34	?	0.27	0.10
	PR	–	40.42	97.11	29.29	0.59	62.39	0.89	33.25	34.08

$\text{SemRank}(1, p) = I(p) \times [1 + R(p)]$, emphasizing unexpected and newly gained information.

The DBPedia PageRank[7] (PR) is an indicator for average 'hub' factor of resources and their neighbourhood based links, how 'common' they are [20].

Table 3 summarizes and explains each of the used measures. Table 4 shows the various improvements of the control algorithm using different measures: both the original and optimized algorithms were configured with the same, the Jaccard distance, weight and heuristic.

5.3 Effect of Weights and Heuristics

The results, shown in Fig. 1, confirm the findings in the detailed sample, but this time the original algorithm uses a combination of the Combined Node Degree (CND) and the Jaccard distance, while the optimized algorithm was configured using a variety of heuristics and weights. To be able to compare the results with each other each of the SR measures are normalized as follows: $\text{SRn} = \frac{\text{SR}}{\max(\text{SR})}$.

[7] http://people.aifb.kit.edu/ath#DBpedia_PageRank.

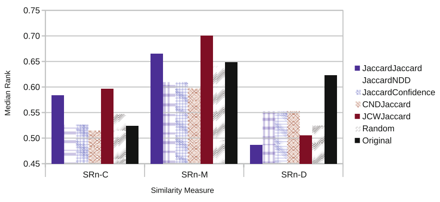

Fig. 1. Effects of the different combinations of weights and heuristics on the measured SemRank. (Color figure online)

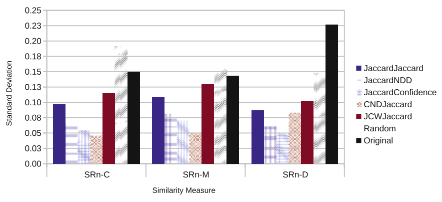

Fig. 2. Standard deviation of the measured SemRank when using different heuristics. (Color figure online)

The standard deviation of the results, shown in Fig. 2, highly differs for each case. In particular when using a random number instead of a weighting function and a heuristic leads to a high standard deviation, which is expected - given the randomness. The deviation is also relatively high when using the Jiang-Conrath distance as weight (JCW) and when using the original algorithm.

On the one hand the conventional and mixed mode for SemRank put less emphasis on novelty and focuses mainly on semantic association and information content. The jaccard distance combination used as weight and heuristic is not entirely surprisingly the best choice for this scenario. On the other hand the results of the original algorithm making use of the common node degree as weight together with the jaccard distance is confirmed by the results of the improved algorithm with the common node degree however with a slightly lower rank in the new algorithm. Using the JCW however leads to even higher ranks. In terms of discovery, the original algorithm outperforms the JaccardJaccard combination. The CNDJaccard improved algorithm is able to slightly outperform all the other combinations.

5.4 User Judgments

We presented the output of each of the algorithms as a list of story facts using the scientists example cases S1–S6 as shown in Table 4, typically 1 up to 20 facts depending on the heuristic that was used. As with SemRank, we are interested in the serendipity as a balance between unexpected facts and relevant facts. We asked the users to rate the list of facts in terms of: (i) relevance; (ii) consistency; and (iii) discovery. The users had to indicate how well the list of facts scored according to them on a Likert scale from -2 (None, Not, Very Poor) to $+2$ (Most, Very, Very Good). A score of 0 (neutral) was only possible in the case of relevance. In total we collected 840 judgments, 20 judgments for each combination of scenario and heuristic. The overall results of the user judgments, rescaled to a score between 0 and 1 are: **relevancy** 0.45; **consistency** 0.45; and **discovery** 0.33. The standard deviations are 0.34; 0.39 and 0.35 respectively. The scores around 0.5 can be interpreted as a disagreement between the users.

The overall score is below 0.5, this indicates that the majority of users judges most of the presented list of story facts below normal or expected relevancy, consistency and with little unexpected new facts. The standard deviation of the user judgments is relatively high, which means that they cover a broad range of judgments some users are very positive while other users are very negative. The mixed results are likely due to varying expectations: some might expected more in-depth results while others appreciated the basic facts about the scientists. The suggested stories that center around a certain via-fact are not always considered relevant by some users even though the algorithms might consider them so. Some examples:

- The users least agreed on the following facts about Carl Linnaeus and Albert Einstein, a score of **0.48** (very little effect) and standard devation of **0.39** when using the JCWJaccard :
```
Carl Linnaeus and Baruch Spinoza are Expert, Intellectual and Scholar
Baruch Spinoza's and Albert Einstein's are both Pantheists Intellectuals and
Jewish Philosophers
```
- The most relevant *and* consistent facts were found between Charles Darwin and Carl Linnaeus: a score of **0.65** and **0.6** respectively with CNDJaccard.
```
Copley Medal's the award of Alfred Russel Wallace and Charles Darwin
Alfred Russel Wallace's and Charles Darwin's awards are Royal Medal and Copley Medal
Alfred Russel Wallace and Charles Darwin are known for their Natural
selection
Carl Linnaeus  and Alfred Russel Wallace have as subject 'Fellows of the
Royal Society'
Carl Linnaeus and Alfred Russel Wallace are Biologists and Colleagues
```
- In terms of discovery the highest score has relatively little agreement among users: **0.48** and standard deviation **0.42** with JCWJaccard:
```
Albert Einstein's and Charles Darwin's reward is Copley Medal.
```

The scores for relevancy, consistency and discovery as unexpected - but relevant - facts are highly dependent on the user who judges. Some users might be interested in the more trivial path as well in some cases. Nevertheless, we used the overall judgment as a baseline to compare the judgments with the same combinations of heuristics and weights as before.

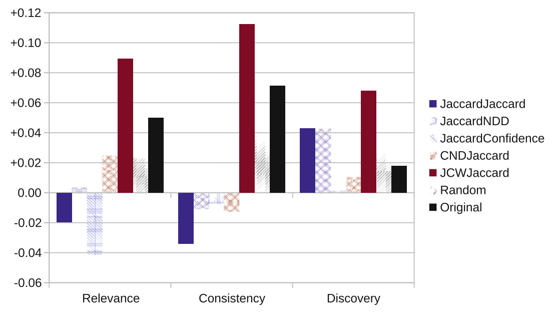

Fig. 3. The effect of the heuristics according to user judgments compared to the overall median. The JCWJaccard confirms already good results with SemRank. The CNDJaccard scores relatively well. (Color figure online)

6 Conclusions and Future Work

We proposed an optimized pathfinding algorithm for storytelling that reduces the number of arbitrary resources revealed in paths contained in the story. Preliminary evaluation results using the DBpedia dataset indicate that our proposal succeeds in telling a story featuring better link estimation, especially in cases where the previous algorithm did not make seemingly optimal choices of links. By defining stories as chains of links in Linked Data, we optimized the storytelling algorithm and tested with several heuristics and weights. The most consistent output was generated with the Jaccard distance used both as weight and heuristic; or as heuristic in combination with the Jiang-Conrath distance as weight. The most arbitrary facts occur in a story when using the combined node degree as weight with the Jaccard distance as heuristic, both in the optimized and the original algorithm. User judgments confirm the findings for the Jiang-Conrath weight and the original algorithm and for the Jaccard distance used as weight and heuristic in terms of discovery. There is no clear positive effect however according the users in terms of consistency and relevancy there.

Future work will focus on validating the correlation between the effect of the link estimation on the arbitrariness as perceived by users and computational semantic relatedness measures such as SemRank. Additionally, we will measure the scalability of our approach by implementing the algorithms (i) solely on the client, (ii) completely on the sever, and (iii) in a distributed client/server architecture.

References

1. Aizawa, A.: An information-theoretic perspective of Tf-idf measures. Inf. Process. Manag. **39**(1), 45–65 (2003)
2. Aleman-Meza, B., Halaschek, C., Arpinar, I.B., Sheth, A.P.: Context-aware semantic association ranking (2003)
3. Aleman-Meza, B., Halaschek-Weiner, C., Arpinar, I.B., Ramakrishnan, C., Sheth, A.P.: Ranking complex relationships on the semantic web. IEEE Internet Comput. **9**(3), 37–44 (2005)
4. Anyanwu, K., Maduko, A., Sheth, A.: Semrank: ranking complex relationship search results on the semantic web. In: Proceedings of the 14th International Conference on World Wide Web, pp. 117–127. ACM (2005)
5. Cheng, G., Zhang, Y., Qu, Y.: Explass: exploring associations between entities via top-k ontological patterns and facets. In: Mika, P., et al. (eds.) ISWC 2014, Part II. LNCS, vol. 8797, pp. 422–437. Springer, Heidelberg (2014)
6. Cilibrasi, R.L., Vitanyi, P.M.: The google similarity distance. IEEE Trans. Knowl. Data Eng. **19**(3), 370–383 (2007)
7. De Meester, B., De Nies, T., De Vocht, L., Verborgh, R., Mannens, E., Van de Walle, R.: StoryBlink: a semantic web approach for linking stories. In: Proceedings of the 14th International Semantic Web Conference (ISWC) Posters and Demonstrations Track (2015)
8. De Vocht, L., Beecks, C., Verborgh, R., Seidl, T., Mannens, E., Van de Walle, R.: Improving semantic relatedness in paths for storytelling with linked data on

the web. In: Gandon, F., Guéret, C., Villata, S., Breslin, J., Faron-Zucker, C., Zimmermann, A. (eds.) The Semantic Web: ESWC 2015 Satellite Events. LNCS, vol. 9341, pp. 31–35. Springer, Heidelberg (2015)

9. De Vocht, L., Coppens, S., Verborgh, R., Vander Sande, M., Mannens, E., Van de Walle, R.: Discovering meaningful connections between resources in the web of data. In: Proceedings of the 6th Workshop on Linked Data on the Web (LDOW2013) (2013)
10. Fang, L., Sarma, A.D., Yu, C., Bohannon, P.: Rex: explaining relationships between entity pairs. Proc. VLDB Endow. **5**(3), 241–252 (2011)
11. Foster, A., Ford, N.: Serendipity and information seeking: an empirical study. J. Doc. **59**(3), 321–340 (2003)
12. Franzoni, V., Mencacci, M., Mengoni, P., Milani, A.: Heuristics for semantic path search in Wikipedia. In: Murgante, B., et al. (eds.) ICCSA 2014, Part VI. LNCS, vol. 8584, pp. 327–340. Springer, Heidelberg (2014)
13. Godin, F., De Nies, T., Beecks, C., De Vocht, L., De Neve, W., Mannens, E., Seidl, T., de Walle, R.V.: The normalized freebase distance. In: Presutti, V., Blomqvist, E., Troncy, R., Sack, H., Papadakis, I., Tordai, A. (eds.) ESWC Satellite Events 2014. LNCS, vol. 8798, pp. 218–221. Springer, Heidelberg (2014)
14. Hart, P., Nilsson, N., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Trans. Syst. Sci. Cybern. **4**, 100–107 (1968)
15. Jiang, J.J., Conrath, D.W.: Semantic similarity based on corpus statistics and lexical taxonomy (1997). arXiv preprint arXiv:cmp-lg/9709008
16. Kumar, D., Ramakrishnan, N., Helm, R.F., Potts, M.: Algorithms for storytelling. IEEE Trans. Knowl. Data Eng. **20**(6), 736–751 (2008)
17. Mazuel, L., Sabouret, N.: Semantic relatedness measure using object properties in an ontology. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 681–694. Springer, Heidelberg (2008)
18. Moore, J.L., Steinke, F., Tresp, V.: A novel metric for information retrieval in semantic networks. In: Proceedings of 3rd International Workshop on Inductive Reasoning and Machine Learning for the Semantic Web (IRMLeS 2011), Heraklion, Greece, May 2011
19. Nies, T.D., Beecks, C., Godin, F., Neve, W.D., Stepien, G., Arndt, D., Vocht, L.D., Verborgh, R., Seidl, T., Mannens, E., de Walle, R.V.: A distance-based approach for semantic dissimilarity in knowledge graphs. In: Proceedings of the 10th International Conference on Semantic Computing (2016, accepted)
20. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: bringing order to the web (1999)
21. Vander Sande, M., Verborgh, R., Coppens, S., De Nies, T., Debevere, P., De Vocht, L., De Potter, P., Van Deursen, D., Mannens, E., Van de Walle, R.: Everything is connected: using linked data for multimedia narration of connections between concepts. In: Proceedings of the 11th International Semantic Web Conference Posters and Demo Track, November 2012
22. Verborgh, R., et al.: Querying datasets on the web with high availability. In: Mika, P., et al. (eds.) ISWC 2014, Part I. LNCS, vol. 8796, pp. 180–196. Springer, Heidelberg (2014)