On the Robustness of Standalone Referring Expression Generation Algorithms Using RDF Data

Pablo Ariel Duboue and **Martin Ariel Domínguez** and **Paula Estrella**
Facultad de Matematica, Astronomía y Física
Universidad Nacional de Córdoba
Córdoba, Argentina

Abstract

A sub-task of Natural Language Generation (NLG) is the generation of referring expressions (REG). REG algorithms are expected to select attributes that unambiguously identify an entity with respect to a set of distractors. In previous work we have defined a methodology to evaluate REG algorithms using real life examples. In the present work, we evaluate REG algorithms using a dataset that contains alterations in the properties of referring entities. We found that naturally occurring ontological re-engineering can have a devastating impact in the performance of REG algorithms, with some more robust in the presence of these changes than others. The ultimate goal of this work is observing the behavior and estimating the performance of a series of REG algorithms as the entities in the data set evolve over time.

1 Introduction

The main research focus in NLG is the creation of computer systems capable of generating human-like language. According to the consensus Natural Language Generation (NLG) architecture (Cahill et al., 2001) the NLG task takes as input non-linguistic data and operates over it as a series of enrichment steps, culminating with fully specified sentences from which output strings can be read out. Such a generation pipeline mimics, up to a certain extent, a Natural Language Understanding (NLU) pipeline. In NLU, however, it is expected that the text upon which the system is being run upon might contain a variety of errors. These errors include wrongly written text that a human might find it difficult to understand or idiosyncratic deviations from well accepted prose (what is called "improper grammar" or "orthographic mistakes" by defenders of prescriptive grammar). The fact that plenty of texts of interest to NLU exhibit poor quality explains the reason behind NLU's focus on *robust* approaches. Such approaches attempt to cope gracefully with inputs that do not conform to the standards of the original texts employed for building the system (either as working examples or training data in a machine learning sense). In NLG, on the other hand, current approaches rarely explore fallback strategies for those cases where the data is not fully compliant with the expected input and, thus, there is little intuition about possible outputs of a system under such circumstances.

In this work we aim to explore robustness for the particular case of Referring Expressions Generation (REG) algorithms by means of different versions of an ontology. Therefore, we can combine REG algorithms with ontologies to study their behavior as the entities in the chosen ontology change. In our case, we have chosen the ontology built from Wikipedia though different versions of DBpedia and three REG algorithms on which we will measure robustness, defined here as an algorithm's resilience to adapt to changes in the data or its capability to gracefully deal with noisy data. In a sense, we are interested in two different phenomena: (1) whether an NLG subcomponent (REG in particular) can be used with outdated ontological data to fulfill its task and (2) which implementation of the said subcomponent is better suited in this setting. Our research is driven by the second question but this current work sheds more light on the first one. See Section 7 for details.

This paper is structured as follows: next section briefly mentions the relevant related work, in Section 3 and Section 4 we describe the algorithms applied and data used, respectively; in Section 5 we describe the setup used in the experiments, then

Section 6 presents the results which are further discussed in Section 7.

2 Related work

The need to account for changes in ontologies has long been acknowledged, given that they may not be useful in real world applications if the representation of the knowledge they contain is outdated. Eder and Koncilia (2004) present a formalism to represent ontologies as graphs that contain a time model including time intervals and valid times for concepts. They base their formalism on techniques developed for temporal databases, namely the versioning of databases instead of their evolution and they provide some guidelines on its possible implementation.

Another source of ontology transformation is spatiotemporal changes. Dealing with spatial changes on historical data (or over time series) is crucial for some NLP tasks, such as information retrieval (Kauppinen and Hyvnen, 2007). In their case, the authors deal with the evolution of the ontology's underlying domain instead of its versioning or evolution due to developments or refinements. Their main result is the definition of partial overlaps between concepts in a given time series, which was applied to build a Finnish Temporal Region Ontology, showing promising results.

More recently, with the development of one of the largest ontologies, DBpedia (Bizer et al., 2009), much research has been devoted to exploiting this resource in NLP or NLG tasks as well as to model its changes. For example, there is research on modeling DBpedia's currency (Rula et al., 2014), that is, the age of the data in it and the speed at which those changes can be captured by any system. Although currency could be computed based on the modification/creation dates of the resources, this information is not always present in Wikipedia pages. To overcome this, the authors propose a model to estimate currency combining information from the original related pages and a couple of currency metrics measuring the speed of retrieval by a system and basic currency or timestamp. Their experiments suggest that entities with high system currency are associated with more complete DBpedia resources and entities with low system currency appear associated with Wikipedia pages that are not easily tractable (or that "could not provide real world information" according with the authors).

Closer to our work, Kutlak et al. (2013) use DBpedia for REG. As opposed to classical work in the field, this work experiments with entities that potentially have a large number of distractors, a situation that may be difficult to handle for classical REG algorithms. Under this hypothesis, the authors propose a new corpus-based algorithm inspired by the notion of Communal Common Ground (CCG), defined as information shared by a particular community whose members assume that everyone knows. In the extrinsic evaluation CCG outperformed the more classical Incremental Algorithm (IA) (Dale and Reiter, 1995), thus authors suggest that CCG might be more suitable for large domains than other algorithms.

In previous work, we (Pacheco et al., 2012) employed several REG algorithms to generate referring expressions from DBpedia, in particular we used Full Brevity (Bohnet, 2007), Constraint Satisfaction approach (Gardent, 2002) and the Incremental Algorithm (Dale and Reiter, 1995). Exploiting articles from Wikinews to generate the RE for a randomly selected group of entities (a technique we also used in this work), we found that DBpedia contained information about half of the entities mentioned in the news and that the IA and the Constraint Satisfaction were able to generate a definite description in about 98% of the contexts extracted from the news articles. However, the algorithms produced satisfactory definite descriptions only in about 40% of the cases. The main problems identified in these experiments were that some properties are quite unique but lead to descriptions of little use to most people evaluating them (thus producing the low results obtained) and the rather odd choice of the preference order of the Incremental Algorithm. Our focus on robustness, however, is different from it.

In an earlier version of this work (Duboue et al., 2015), we analyzed only people rather than people and organizations and had an experimental mishap where the old version included tuples from the new version of DBpedia, producing results that were too optimistic. Even with the issue with the previous results, it encouraged us to use robustness as a metric for learning the IA ordering, an intriguing idea we explored recently, with mixed results (Duboue and Domínguez, 2016).

3 Algorithms

In this section we describe the REG algorithms used in this work. We chose three representative algorithms that can deal with single entity referents, a classic algorithm (Dale and Reiter, 1995), an algorithm generating negations (Gardent, 2002) and an algorithm using graph theory (Krahmer et al., 2003). We describe each of the algorithms using the following notation: R is the referent, C is the set of distractors and P is a list of properties, triples in the form (entity, attribute, value), describing R.

REG algorithms pick properties that might ultimately be used to generate nominal syntactic units that identify the entity that is the referent.[1] We define the context as the set of entities that the receiver is currently paying attention to. Then, the distractor set is the context set without the predicted reference. Once this is defined, they consider the components of a referral expression as rules to set aside or keep on considering the members of the contrast set. For example, if the speaker wants to identify a *small black dog* in a situation where the distractor set consists of a *big white dog* and a *black small cat*, we could choose the adjective black to rule out the white dog and then the noun dog to rule out the noun cat. The resulting referral expression would be the black dog, which refers to R but not to the other entities in this context. Thus, the R was unmistakably identified.

The algorithms listed here are implemented in the Java programming language and publicly available under an Open Source license as part of the Alusivo REG project.[2]

3.1 Incremental Algorithm

The incremental algorithm assumes the properties in P are ordered according to an established criteria. Then the algorithm iterates over P, adding each triple one at a time and removing from C all entities ruled out by the new triple. Triples that do not eliminate any new entity from C are ignored. The algorithm terminates when C is empty. This algorithm was created in 1995 (Dale and Reiter, 1995) as a simplification of previous work on the development of REG algorithms. Given its simplicity it is considered a baseline in many NLG articles.

This algorithm is strongly influenced by the preference order among attributes. We used the ordering for people in Wikipedia developed by Pacheco (Pacheco, 2012) which is shipped with Alusivo.

3.2 Gardent Algorithm

The algorithm (Gardent, 2002) is based on the idea that in many languages, a possible way to unambiguously describe entities is to identify a set of related referents and to provide a quantified expression, for example, *"the team where Messi has played that is not in Argentina"* should suffice to identify *Barcelona Ftbol Club* as the referred entity. The speaker offers enough information to the listener to identify the set of objects the speaker is talking about. From a generation perspective, this means that starting with a set of objects and their properties which are known to the speaker and the listener, a distinctive description must be created, in such a way that it allows the user to unmistakably identify the referred objects. The solution addressed from this standpoint is an algorithm that generates minimal distinct descriptions, that is to say, with the least number of literals to identify the target. By definition, these will not be unnecessarily long, redundant nor ambiguous. The algorithm performs this task using Constraint Satisfaction Programming (CSP) (Lassez, 1987)[3] to solve two basic constraints: find a set of positive properties P^+ and negative properties P^-, such that all properties in P^+ are true for the referent and all in P^- are false, and it is the smaller $P^+ \cup P^-$ such that for every $c \in C$ there exists a property in P^+ that does not hold for c or a property in P^- that holds for c.

We further reuse the orderings from the incremental algorithm for the search process used by the constraints solver.

3.3 Graph Algorithm

The REG graph-based algorithm (Krahmer et al., 2003) constructs an initial directed graph G that models the original problem. The nodes in G belong to $\{R\} \cup C$ and the edges represent the properties P. The algorithm recursively constructs a

[1] At this level in the generation pipeline, the system operates on abstract semantic representations, the actual words and syntactic forms are left to other components, such as the lexical chooser and the surface generator. In this discussion we use nominal phrases to illustrate the topics, but this is with the understanding that is the output of the full system not the REG algorithm alone.

[2] `https://github.com/DrDub/Alusivo`

[3] We employed the Choco CSP solver Java library: `http://choco-solver.org/`.

sub-graph of G, V, starting with node R. At each step it explores the space of successors of the nodes in G that are not in V. The search is guided by a cost function that is calculated each time a node and edge are added to V. The goal of the algorithm is to check whether the properties in V serve to distinguish R from C, meaning that V is distinctive. To verify whether V is distinctive, at each step the algorithm searches for the existence of a sub-graph G_c isomorphic to V, where G_c contains a node $c \in C$; if no such graph exists, then V is distinctive. Finally, the least expensive distinctive graph is returned.

Different cost functions are possible. For our experiments, we use a cost function equal to the number of edges plus the number of vertices in the subgraph and we order the search over edges using the same ordering as the incremental algorithm. This is the particular parameterization of the graph algorithm that we are comparing against the other algorithms.

4 Data

We have chosen Wikipedia as our source of entities, as it represents one of the biggest freely available knowledge base. Started in January 2001 at present it contains over 37 million articles in 284 languages and it continues to grow thanks to the collaborative creation of content by thousands of users around the globe.

Given that the content in Wikipedia pages is stored in a structured way, it is possible to extract and organize it in an ontology-like manner as implemented in the DBpedia community project. This is accomplished by mapping Wikipedia infoboxes from each page to a curated shared ontology that contains 529 classes and around 2,330 different properties. DBpedia contains the knowledge from 111 different language editions of Wikipedia and, for English the knowledge base consists of more than 400 million facts describing 3.7 million things (Lehmann et al., 2015). A noble feature of this resource is that it is freely available to download in the form of *dumps* or it can be consulted using specific tools developed to query it.

These dumps contain the information coded in a language called Resource Description Framework (RDF) (Lassila et al., 1998). The WWW Consortium (W3C) has developed RDF to encode the knowledge present in web pages, so that it is comprehensible and exploitable by agents during any information search. RDF is based on the concept of making statements about (Web) resources using expressions in the subject-predicate-object form. These expressions are known as triples, where the subject denotes the resource being described, the predicate denotes a characteristic of the subject and describes the relation between the subject and the object. A collection of such RDF declarations can be formally represented as a labeled directed multi-graph, naturally appropriate to represent ontologies.

We have chosen to use the dumps of different versions of Wikipedia, namely versions 2014 (09/2014) and 3.6 (01/2011). DBpedia 3.6 ontology encompasses 359 classes and 1,775 properties (800 object properties, 859 datatype properties using standard units, 116 datatype properties using specialized units) and DBpedia 2014 ontology encompasses 685 classes and 2,795 properties (1,079 object properties, 1,600 datatype properties using standard units, 116 datatype properties using specialized units).[4] These versions have been specifically selected: the 2014 version for current up-to-date data and the 3.8 version for comparison with the results by Pacheco et al. (Pacheco et al., 2012).

5 Experimental setup

We follow an approach similar to Pacheco et al. (Pacheco et al., 2012) to extract REG tasks from journalistic text: we extract all people that appear explicitly linked in a given Wikinews article. By using Wikinews, we ensure all the people are disambiguated to their DBpedia URIs by construction (Figure 1).[5] We selected a Wikinews dump as closest to our target DBpedia (20140901). From there, we define all URIs for which DBpedia has a birthDate relation (761,830 entities) as "people" and all entities with a foundDate as an "organization" (19,694 entities). We extract all such people and organizations that appear in the same Wikinews article using the provided inter-wiki SQL links file. For each article, we randomly chose a person as the referent, turning them into a fully defined REG task. This approach produced 4,741 different REG tasks, over 9,660 different people and 3,062 over 8,539

[4] Statistics taken from the DBpedia change log available at http://wiki.dbpedia.org/services-resources/datasets/change-log.

[5] These are *potential* REG tasks, but not *actual* REG tasks. We use the news article to extract naturally co-occurring entities.

Algorithm	Execution Errors	Dice	Omission Errors	Inclusion Errors
People				
Incremental	232 (5%)	0.48	1,406 (50%)	145 (5%)
Gardent	0 (0%)	0.58	1,089 (36%)	554 (18%)
Graph	15 (0%)	0.38	1,870 (62%)	20 (0%)
Organizations				
Incremental	1,386 (45%)	0.69	305 (31%)	3 (0%)
Gardent	829 (27%)	0.70	338 (22%)	357 (23%)
Graph	934 (31%)	0.06	1,347 (94%)	2 (0%)

Table 1: Our results, over 3,051 different REG tasks for people and 2,370 for organizations. The error percentages are computed over the total number of executed tasks.

organizations. Each REG task has an average of 4.89 people and 2.79 organizations.

We then created a subset of the relevant tuples for these people (291,039 tuples on DBpedia 2014 and 129,782 on DBpedia 3.6, a 224% increase[6]) and organizations (468,041 tuples on DBpedia 2014 and 216,730 on DBpedia 3.6, a 216% increase) by extracting all tuples were any of the people or organizations were involved, either as subject or object of the statement. Over these subsets were our algorithms executed.

As we are interested in REs occurring after first mentions, we filter properties from the data that unequivocally identify the entity, such as full name or GPS location of its headquarters.

6 Results

We run three representative algorithms that can deal with single entity referents, described in Section 3. As all our algorithms can fail to produce a RE, the first column in our results table (Table 1) contains the number of execution errors. The algorithms can fail either because they have a timeout argument (such as in Graph) or they have a restricted set of heuristics (like IA) that might fail to produce a unique description. In the event of unknown entities (36% of people tasks and 23% of organization tasks contain an entity unknown to DBpedia 3.6), Gardent Algorithm can attempt to produce a RE using negations (using a closed world assumption) while our implementation of the Graph Algorithm will also attempt building a RE if the unknown entity is not the referent. This seldom happens and we report only numbers on fully defined tasks, which mean the algorithms can be run only on 64% of people tasks and 77% of organization tasks. Using the RE obtained in the old version of DBpedia, we executed on the new version and computed a Dice set similarity coefficient between the two sets. However, Dice has its own problems when it comes to evaluating REG results (van Deemter and Gatt, 2009) and we thus computed two extra metrics: inclusion errors and omission errors. Inclusion errors imply the RE chosen on the old version of DBpedia included extra distractors when applied to the new version of DBpedia. Omission errors imply the RE chosen on the old version of DBpedia failed to include the referent in the new version (this figure includes all execution errors).

The number of inclusion errors is somewhat in line with our expectations from having a more detailed ontological resource: as more data is known about the world, properties that seemed a good fit in a knowledge poor situation all of sudden become too general. For example, trying to distinguish a politician A from two other politicians B, C when we do not have a record that B and C are politicians will lead us to a RE (*x is a politician*) that will overgenerate on a newer, richer ontology.

The number of omission errors was, however, puzzling at first sight and wholesome unacceptable. A certain level of omission errors can be expected (referring to a former prime minister as a prime minister will result in an omission error) but a 3 year span cannot justify 50% omission errors. Further error analysis reveals two key changes in DBpedia that result in this behavior: a re-engineering of its type system (dropping large number of types, such as 'Politi-

[6]In DBpedia 2014, there was an average of 30.12 properties per person while in DBpedia 3.6, there was an average of 17.3

cian') and dropping language annotations for values (from "22nd"@en to "22nd" –note the dropping of @en). These two changes account for 90% of the omission errors and even a greater percentage of the errors produced by the Graph Algorithm in organizations (the 0.06 Dice in the table).

Now, with further engineering on our behalf (or by choosing a different version of DBpedia where these changes have been already ironed out), we can have an experiment that will shed more light over which REG algorithm is more robust. However, for the sake of our first research question, we find these results quite enlightening: these ontological changes were difficult to spot (the new version was 200% *bigger*, there were little reason to expect many types were *dropped*). Moreover, type information is key for most REG algorithms. We believe this highlights a different point-of-view when designing NLG subcomponents that operate with real data in a changing world.

Given this, our results are too early to fully compare REG algorithms but as a preliminary result, the CSP approach seems to perform consistently at the top. We also found a few interesting examples presented in the appendix.

All our code and data are available online for further analysis.[7]

7 Conclusions

We set ourselves to investigate the robustness of a NLG subcomponent when applied to Web data in RDF format. We were interested in two different phenomena: (1) whether an NLG subcomponent (REG in particular) can be used with outdated ontological data to fulfill its task and (2) which implementation of said subcomponent is better suited to this setting. Our research is driven by the second question but this current work sheds more light on the first one: we found that, off the bat, one quarter of the entities of interest, using three year old data, will be unknown. Handling unknown ontological entities is a problem that has received no attention in NLG as far as we can tell (compare this to dealing with OOV words in NLU, a well defined task and problem). Moreover, we found that in the particular span we have chosen, the typesystem underwent massive re-engineering, which in turn renders the old referring expressions meaningless for this exercise[8] (and renders their associated resources, such as lexicons, stale). Given these errors, it is still too early to conclude which algorithm from the three REG algorithms we analyzed fares better in this setting, but we found early evidence in favor of the constraint satisfaction algorithm proposed by Gardent (2002). We also believe that there is space for a new REG algorithm design with resiliency in mind that seeks to produce REs that hold better over time.

```
Former [[New Mexico]] {{w|Governor of New
Mexico|governor}} {{w|Gary Johnson}} ended
his campaign for the {{w|Republican Party
(United States)|Republican Party}} (GOP)
presidential nomination to seek the backing
of the {{w|Libertarian Party (United
States)|Libertarian Party}} (LP).
```

Figure 1: Wikinews example, from `http://en.wikinews.org /wiki/U.S. _presidential _candidate _Gary _Johnson _leaves _GOP _to _vie _for _the _LP _nom`, adapted from Pacheco et al. (Pacheco et al., 2012).

Our comparison has been done over three specifically parameterized versions of the chosen algorithms. We cannot conclude whether the differences among them are due to differences in the algorithms themselves or in their parameterizations. We believe a follow-up study measuring the impact of different parameterizations in this setting is merited.

Also in future work, we plan to simulate natural perturbations on the data in order to find the conditions on which REG algorithms start to fail (for example, a simulated DBpedia of 25 years in the future).

Acknowledgments

We would like to thank the anonymous reviewers as well as Annie Ying and Victoria Reggiardo.

References

C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. 2009. DBpedia-a crystallization point for the web of data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(3):154–165.

B. Bohnet. 2007. is-fbn, is-fbs, is-iac: The adaptation of two classic algorithms for the generation of referring expressions in order to produce expressions like

[7] `https://github.com/DrDub/Alusivo` and `https://duboue.net/data`.

[8] Whether or not the RE still hold in the real world is a topic we did not investigate, given the size of our evaluation.

humans do. *MT Summit XI, UCNLG+ MT*, pages 84–86.

Lynne Cahill, John Carroll, Roger Evans, Daniel Paiva, Richard Power, Donia Scott, and Kees van Deemter. 2001. From rags to riches: exploiting the potential of a flexible generation architecture. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 106–113. Association for Computational Linguistics.

R. Dale and E. Reiter. 1995. Computational interpretations of the gricean maxims in the generation of referring expressions. *Cognitive Science*, 19(2):233–263.

Pablo Duboue and Martín Domínguez. 2016. Using robustness to learn to order semantic properties in referring expression generation. In *Proceedings of Iberamia 2016 (to appear)*.

Pablo Duboue, Martín Domínguez, and Paula Estrella. 2015. Evaluating robustness of referring expression generation algorithms. In *Proceedings of Mexican International Conference on Artificial Intelligence 2015*. IEEE Computer Society.

Johann Eder and Christian Koncilia. 2004. C.: Modelling changes in ontologies. In *In: Proceedings of On The Move - Federated Conferences, OTM 2004, Springer (2004) LNCS 3292*, pages 662–673.

C. Gardent. 2002. Generating minimal definite descriptions. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 96–103. Association for Computational Linguistics.

Tomi Kauppinen and Eero Hyvnen. 2007. Modeling and reasoning about changes in ontology time series. In *Integrated Series in Information Systems*, pages 319–338. Springer-Verlag.

Emiel Krahmer, Sebastiaan Erk, and Andr Verleg. 2003. Graph-based generation of referring expressions. *Computational Linguistics*, 29:53–72.

Roman Kutlak, Kees van Deemter, and Chris Mellish. 2013. Generation of referring expressions in large domains. *PRE-CogSci, Berlin*.

Jean-Louis Lassez. 1987. Logic programming. In *Proceedings of the Fourth International Conference*.

Ora Lassila, Ralph R. Swick, World Wide, and Web Consortium. 1998. Resource description framework (rdf) model and syntax specification.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 6(2):167–195.

Fabián Pacheco, Pablo Ariel Duboue, and Martín Ariel Domínguez. 2012. On the feasibility of open domain referring expression generation using large scale folksonomies. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12, pages 641–645, Stroudsburg, PA, USA. Association for Computational Linguistics.

Fabián Pacheco. 2012. Evaluacin en gran escala de algoritmos clsicos parade expresiones referenciales generacin. Master's thesis, Facultad de Matematica Astronomia y Fisica, Universidad Nacional de Cordoba.

Anisa Rula, Luca Panziera, Matteo Palmonari, and Andrea Maurino. 2014. Capturing the currency of dbpedia descriptions and get insight into their validity. In *Proceedings of the 5th International Workshop on Consuming Linked Data (COLD 2014) co-located with the 13th International Semantic Web Conference (ISWC 2014), Riva del Garda, Italy, October 20, 2014*.

Kees van Deemter and Albert Gatt. 2009. Beyond dice: measuring the quality of a referring expression. In *Proceedings of the Workshop on Production of Referring Expressions: Bridging Computational and Psycholinguistic Approaches*.

Figure 2: Appendix: Selected Runs

- Distinguish *Saddam Hussein* from *Paul Volcker, Kofi Annan, Boutros Boutros-Ghali*

  Incr orderInOffice: "President of Iraq"
  Gardent orderInOffice: "President of Iraq"
  Graph activeYearsEndDate: 2006-12-30

  All algorithms perform well.

- Distinguish *Daniel Vettori* from *Kyle Mills*

  Incr country: New Zealand
  Gardent NOT country: New Zealand national cricket team
  ***Graph** description: "New Zealand cricketer"

  Here the accuracy of the information comes into play. Both people are New Zealand cricketers but the information on Mills is poorer than on Vettori. The REs for all algorithms are incorrect but work due to lack of data on Mills. In the new version of DBpedia the description attribute for Mills has been added and now Graph fails. The other two algorithms still work well, even if they should not.

- Distinguish *Park Geun-hye* from *Martin Dempsey*

  Incr type: Office Holder
  Gardent NOT type: Military Person
  ***Graph** birth year: 1952

  This is very curious, both people are born in the same year, but that information was missing in the old version of DBpedia for the distractor.

- Distinguish *Paul McCartney* from *Ringo Starr*, *John Lennon*, *George Harrison*

  Incr instrument: Hfner 500/1
  Gardent NOT associated musical artist: Plastic Ono Band
  ***Graph** background: solo singer

  In the old DBpedia, McCartney was the only Beatle marked as a solo singer, while in the new version all of them are. Note how Gardent picks having not played in the Plastic Ono Band as McCartney's most distinguishing feature from the rest.

- Distinguish *Vazgen Sargsyan* from *Karen Demirchyan, Paruyr Hayrikyan, Serzh Sargsyan, Hovik Abrahamyan*

  Incr type: Office Holder, Politician, Prime Minister
  Gardent type: Prime Minister
  ***Graph** death date: 1999-10-27

  Both Sargsyan and Demirchyan died in a tragic shooting at the Armenian parliament. That information was not recorded in the old DBpedia for Demirchyan, leading to the error.