ONTOSPREAD: Activation of Concepts in Ontologies through the Spreading Activation algorithm

Jose María Álvarez[2], Diego Berrueta[1], Luis Polo Paredes[1], and José Emilio Labra Gayo[2]

[1] Fundación CTIC, Gijón, Asturias, Spain,
{diego.berrueta,luis.polo}@fundacionctic.org,
WWW home page: http://www.fundacionctic.org
[2] WESO RG, Universidad de Oviedo, {UO1548,labra}@uniovi.es,
WWW home page: http://www.uniovi.es

Abstract. The present article introduces the ONTOSPREAD API for the development, configuration, customization and execution of the Spreading Activation techniques in the field of the Semantic Web. These techniques have been used to the efficient exploration of knowledge bases based on semantic networks in Information or Document Retrieval areas. ONTOSPREAD enables the implementation of the Spreading Activation algorithm on RDF models and datasets, implicit graph structures. It implements the process of activation and spread of concepts in ontologies applying different restrictions like weight degradation according to the distance or the converging paths reward. The main application of Spreading Activation lies in two different areas: 1) construction of hybrid semantic search engines 2) ranking of resources according to an input set of weighted resources. Finally an evaluation methodology and an example using the Galen ontology are provided to validate the goodness and the capabilities of the proposed approach.

1 Introduction

1.1 Main Contributions

2 Background

In this section, the theoretical model of *SA* is reviewed to illustrate the basic components and the operations performed by the SA techniques during their execution. This model is made up of a conceptual network of nodes connected through relations (conceptual graph). Taking into account that nodes represent domain objects or classes and edges relations among them, it is possible take them as a semantic network in which SA can be applied. The processing performed by the algorithm is based on a thorough method to go down the graph using an iterative model. Each iteration is comprised of a set of beats, a stepwise method, and the checking of a stop condition. Following [8] the basic definitions made are presented:

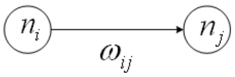

Fig. 1. Graphical model of *Spreading Activation*

***Preadjustement*:** This is the initial and optional stage. It is usually in charge of performing some control strategy over the target semantic network.

***Spreading*:** This is the spread stage of the algorithm. Concepts are activated in activation waves. The spreading node activates its neighbour nodes.

The calculation of the activation rank I_i of a node n_i is defined as follows:

$$I_i = \sum_j O_j \omega_{ji} \tag{1}$$

I_i is the total inputs of the node n_i, O_j is the output of the node n_j connected to n_i and ω_{ji} is the weight of the relation between n_j and n_i. If there is not relation between n_j and n_i then $\omega_{ji} = 0$.
The activation function f is used to evaluate the "weight" of a node and decide if the concept is active.

$$N_i = f(I_i) = \begin{cases} 0 & \text{if } I_i < \jmath_i \\ 1 & \text{if } I_i > \jmath_i \end{cases} \tag{2}$$

N_i is 1 if the node has been activated or 0 otherwise. $\jmath_i$, the threshold activation value for node i, depends on the application and it can change from a node to others. The activation rank I_i of a node n_i will change while algorithm iterates.

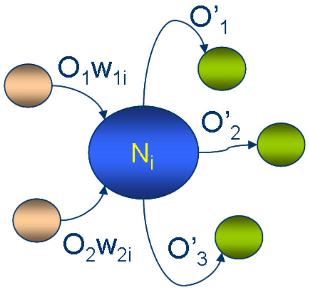

Fig. 2. Activation of the concepts in *Spreading Activation*

***Postadjustment*:** This is the final and optional stage. As well as *Preadjustment* stage, it is used to perform some control strategy in the set of activated concepts.

3 Related Work

4 ONTOSPREAD

4.1 Constrained Spreading Activation

One of the main features of the SA techniques are their flexibility to fit to the resolution of different problems. From the configuration point of view some constraints presented in [7] have been customized improving the expected outcomes of the execution according to the domain problem. Following these constraints are defined:

Distance: nodes far from an activated node should be penalized due to the number of needed steps to reach and activate them.

Path: the activation path is built by the activation process from a node to other and can be guided according to the weights of relations (edges).

Multiple outputs (Fan-Out): nodes "highly connected" can guide to a misleading situation in which activated and spread nodes are not representative, these nodes should be skipped of penalized by the algorithm.

Threshold activation: a node n_i will be spread *iif* its activation value, I_i, is greater than a threshold activation constant $\jmath$.

The abovementioned theoretical model in Sect.**??** is a good start point to design an API for *SA* but from the domain expert point of view some requirements are missing (FIXME: ref) to provide a configurable set of techniques based on *SA*. That is why a set of extensions are proposed to deal with the specific features of ontologies and RDF graphs.

Context of activation $\mathbb{D}_{com}$: concepts can be defined in different domains or schemes. The double process of activation and spreading will only be carried out in the context $\mathbb{D}_{com}$.

Definition 1. *Let $\mathbb{D}_{com}$ an active domain, if a concept c_i is activated o spread then $c_i \in \mathbb{D}_{com}$.*

Minimum activation value $N_{\min}$: only concepts with an activation value N_k greater than $N_{\min}$ will be spread. This constraint comes from the theorical model of *SA*.

Maximum number of spread concepts $\mathbb{M}$: the process of activation and spreading will be executed, at the most, until $\mathbb{M}$ concepts had been spread.

Minimum number of spread concepts $\mathbb{M}_{\min}$: the process of activation and spreading will be executed, at least, $\mathbb{M}_{\min}$ concepts had been spread.

Time of activation t: the process of activation and spreading will be executed, at the most, during t units of time.

Output Degradation O_j: one of the keypoints to improve and customize the algorithm is to define a function h that penalizes the output value O_j of a concept c_j.

1. Generic customization: h calculates the output of a concept c_j according to its degradation level.

$$O_j = h(I_j) \tag{3}$$

Basic case: if $h_0 = id$, the output value O_j takes the level of the activated concept c_j as its value.

$$O_j = h_0(I_j) = I_j \tag{4}$$

2. Customization using **distance**: h_1 calculates the level activation of the concept c_j according to the distance from the initial concept $c_l \in \Phi$[3] that has activated it. The activation value have to decrease if the distance from Φ grows thus the algorithm follows a path from c_l to c_j: $I_l > I_j$. The function h_1 penalizes the output of the concepts (decreasing their rank) far from the "activation core" and rewards closed concepts. Thus, let d_j, where $d_j = min\{d_{lj} : \forall n_l \in \Phi\}$:

$$O_j = h_1(I_j, d_j) = \frac{I_j}{d_j} \tag{5}$$

3. Customization using **beats**: the function h_2 calculates the degradation of the concept using the number of iterations k:

$$O_j = h_2(I_j, k) = (1 + \frac{I_j}{k}) \exp(-\frac{I_j}{k}). \tag{6}$$

4.2 Design of Spreading Activation

The entry point toSA techniques is the set of initial concepts (Q_{sem}) that will generate a new set of the most relevant concepts (Q'_{sem}). Ontologies are a graph where each node n_i represents a concept c_i and the edge ω_{ji} is the semantic relation between c_j y c_i. The final result of the algorithm is a set of sorted pairs (n_i, I_i) that build Q'_{sem}, where $n_i \approx c_i$ and $I_i \approx w_i$ (the relevance of the concept).

The implementation of SA, see Algorithm.4.2, comprises of two sets of concepts that store information about the state of the algorithm: 1) $\mathbb{D}_{com}$ are all the concepts in the semantic network and 2) Φ[4] is the set of initial activated concepts, c_j^k is the spreading concept at the k-esima iteration (from which other concepts are activated).

Set $\mathcal{A}$: queue of **activated** concepts (candidates to be spread).

$$\mathcal{A}^0 = \Phi \tag{7}$$

$$\mathcal{A}^k = (\mathcal{A}^{k-1} \cup \{c_i : \forall c_i / \omega_{ji}^k > 0\}) - \{\mathcal{G}^k\} \tag{8}$$

[3] Set of initial concepts.

[4] $\Phi \equiv Q_{sem}$.

Set $\mathcal{G}$: set of spread concepts:

$$\mathcal{G}^0 = \emptyset \tag{9}$$

$$\mathcal{G}^k = \mathcal{G}^{k-1} \cup \{c_j^k\} \tag{10}$$

The output of the algorithm is the new enriched query $\mathcal{G}^k = Q'_{sem}$ made up of the set of weighted concepts.

Finally, the calculus of the activation value of a concept c_i at iteration k, indicated by I_i^k, is defined. At 0 iteration the activation value c_i is calculated as follows:

$$I_i^0 = \begin{cases} 1 & \text{si } c_i \in \Phi \\ 0 & \text{si } c_i \notin \Phi \end{cases} \tag{11}$$

at k iteration, the activation value of c_i from element c_j^k that activates c_i is calculated as follows:

$$I_i^k = \begin{cases} I_i^{k-1} & \text{si } \omega_{ji}^k = 0 \\ I_i^{k-1} + \omega_{ji}^k I_j^{k-1} & \text{si } \omega_{ji}^k > 0 \end{cases} \tag{12}$$

Algorithm 1 *Pseudocode of Spreading Activation*

```
Require: $\Phi \neq \emptyset$
Ensure: $\mathcal{G} \neq \emptyset$
  $\mathcal{A} \leftarrow \Phi$
  $\mathcal{G} \leftarrow \emptyset$
  while $\mathcal{A} \neq \emptyset$ AND $card(\mathcal{G}) < \mathcal{G}_{\min}$ AND $N_k \geq N_{\min}$ do
    $n_k \leftarrow extract(\mathcal{A})$
    $\mathcal{G} \leftarrow \{n_k\} \cup \mathcal{G}$
    for all $n_i / w_{ki} > 0$ do
      $N_i \leftarrow N_i + w_{ki} N_k$
      $\mathcal{A} \leftarrow (\{n_i\} \cup \mathcal{A}) - \mathcal{G}$
    end for
  end while
  return $\mathcal{G}$
```

4.3 Improving Spreading Activation

Some improvements in the calculus of the activation value of a concept have been introduced in order to get FIXME. If some paths of activation converge to the same node and the source nodes are different then this node should be important and a reward is applied to the nodes presented in these paths.

Definition 2. *Let p_i the number of paths that start and finish in different nodes of Φ*[5] *and they go through the node c_i and they only contain nodes belonging to*

[5] The reward is not applied to nodes in Φ.

$\mathcal{G}$. This improvement assigns a new value to the activation value of each node c_i indicated by I_i^* and it is calculated by means of the function g:

$$I_i^* = g(I_i, p_i) \tag{13}$$

In this case, a relaxed reward function has been chosen, Eq. 14, and, it is applied in the *Postadjustment* stage, thus the original semantics and behaviour of *SA* algorithm is not broken.

$$g(x, y) = x(\log(y + 1) + 1) \tag{14}$$

x is the reward constant, it can be defined according to the context and y is the number of times that a concept c_i must be rewarded.

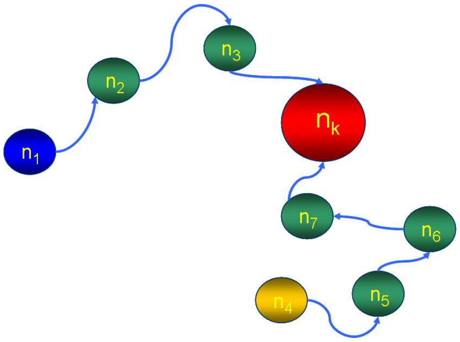

Fig. 3. Paths and rewards in *Spreading Activation*

4.4 Refining Spreading Activation

The whole configuration of the algorithm can be made by default but a customization to a particular domain should be performed by a domain expert taking into account the specific issues of this domain and considering as a new stage of the ontology o graph modelling process.

1. The algorithm is highly coupled to the target ontology and domain. Thus the adjustment and customization of the algorithm should be created or supervised by experts with domain knowledge.
2. The establishment of relation weights among concepts is the key point to customize *SA* techniques.

Since *SA* uses the weights of relations to calculate the activation value of the concepts, different “patterns”, see Table 1, have been identified to manage the direction of the spreading process.

These control patterns can be put together in order to fit as possible the focus and direction of the double process of activation and spreading.

Spread Direction	Definition	Key Relation
Ascending	It seeks for the activation of concepts more generic than the current.	“superclass”
Descending	It seeks for the activation of concepts more specific than the current.	“superclass”
Nominal	It seeks for the activation of instances instead of concepts.	“instance of”
Cross	It seeks for the activation of concepts and instances connected through a certain relation $\mathcal{R}$.	$\mathcal{R}$

Table 1. Patterns of direction control in *SA*.

4.5 Design and implementation of ONTOSPREAD API

ONTOSPREAD API is addressed by an open design and the application of best practices on software design [11,2] and development [?,10]. patterns. The next basic objects need to implement *SA* techniques have been identified.

- The *Player* class handles the execution of the algorithm in a stepwise way. This is an application of the *Iterator* design pattern to the activation and spreading processes. The state of the algorithm is captured in a separate class, *OntoSpreadState* thus it is possible to serialize the state and back to a previous one.
- The *SA* process comprises of three sub-processes:
  1. *Preadjustement*: *OntoSpreadPreAdjustment*; 2. *Spreading* (activation and spreading) with constraints: *OntoSpreadRun*; and 3. *Postadjustment*: *OntoSpreadPostAdjustment*.
  Moreover, the process carries on the information about the knowledge base using the DAO pattern thus the API is independent from the modelling language of the semantic network. Currently, OWL and RDF are supported.

Fig. 4. ONTOSPREAD Overview Diagram

4.6 Designing of the state for *SA*

The keypoint to design the algorithm lies in how and where the information will be available at different iterations. Secondly, an unique entry points to the state of the algorithm should be available trying to avoid illegal accesses to the states. This object stores the next information: 1. Spread concepts. 2. Active concepts. 3. Paths of activation. 4. Concept to be spread. 5. Generic swap area (to share information).

4.7 Designing of restrictions for *SA*

The extensibility and flexibility of the algorithm is subjected to a good design of the restrictions and to the procedure of their evaluation. The next features and design patterns are used to design and implement the model of restrictions for SA:

- Any restriction can be considered as a simple restriction and can be evaluated to a boolean value.
- Conditions or actions in the algorithm can be comprised of several restrictions.
- The extension points of the algorithm, included through a *Template Method* design pattern, are strategies to carry out an specific action. Each strategy can be subjected to one or several restrictions.
- Each restriction can be simple or comprised of others. *Composite* pattern.
- Each action is an strategy. *Strategy* pattern.
- One strategy implies one restriction (or a set of them) thus the strategy is a client of the *Composite* of restrictions.
- The evaluation of the restrictions to get their value (boolean) is carried out through a *Visitor* pattern that fits perfectly to evaluate and walk in composite objects.
- The evaluation process consists on: 1. Apply the strategy, this step modifies the execution and reporting of batch tests. It provides a framework to configure, combine and load several configurations for *SA* and obtain results. We have designed a XML vocabulary using XML-Schema and the *Extensible Content Model* xml design pattern to build the configuration of the *SA* process. The designing of this vocabulary is oriented to be used with JAXB, this technology allow us the generation of Java classes automatically and we can marshalling and unmarshalling objects providing a good way to configure, load and serialize different configurations. The main goal to define a new XML vocabulary can be arguable, but this vocabulary is not a new XML to be interchanged among applications, we only use inside OntoSpreadTest and to provie state of the algorithm. 2. Validate the changes, the restricctions assert the changes.

4.8 Use Cases and Scenarios

5 Evaluation of ONTOSPREAD API

5.1 ONTOPSREAD API in Action

6 Conclussions

6.1 Future Work

Formal[20,7] Data mining[25] Information Retrieval[8,3,1,17] Concept exploration[21] and ontologies[6,18,9,19] Annotations[15,5] Tagging[13] Web Search[28] Natural Language[26] Recommendations[12,16] Semantic Search[24,27,23,4,22,14] Ing. software[2]

References

1. Maristella Agosti and Fabio Crestani. A Methodology for the Automatic Construction of a Hypertext for Information Retrieval. In *SAC '93: Proceedings of the 1993 ACM/SIGAPP symposium on Applied computing*, pages 745–753, New York, NY, USA, 1993. ACM Press.
2. Deepak Alur, John Crupi, and Dan Malks. *Core J2EE Patterns: Best Practices and Design Strategies.* Sun Microsystems, 2003.
3. Helmut Berger, Michael Dittenbach, and Dieter Merkl. An adaptive information retrieval system based on associative networks. In *CRPIT '04: Proceedings of the first Asian-Pacific conference on Conceptual modelling*, pages 27–36, Darlinghurst, Australia, Australia, 2004. Australian Computer Society, Inc.
4. Diego Berrueta, Jose Emilio Labra, and Luis Polo. Searching over public administration legal documents using ontologies. In *Proceedings of Joint Conferente On Knowledge-Based Software Engineering (JCKBSE 2006)*, pages 167 – 175, July–August 2006.
5. Abon Chen, Hsin-Hsi Chen, and Polly Huang. Predicting social annotation by spreading activation. In *Proceedings of the 10th international conference on Asian digital libraries: looking back 10 years and forging new frontiers*, ICADL'07, pages 277–286, Berlin, Heidelberg, 2007. Springer-Verlag.
6. H. Chen and T. Ng. An Algorithmic Approach to Concept Exploration in a Large Knowledge Network (automatic thesaurus consultation): Symbolic Branch-and-Bound search vs. connectionist Hopfield net activation. *J. Am. Soc. Inf. Sci.*, 46(5):348–369, 1995.
7. Paul R. Cohen and Rick Kjeldsen. Information Retrieval by Constrained Spreading Activation in Semantic Networks. *Inf. Process. Manage.*, 23(4):255–268, 1987.
8. F. Crestani. Application of Spreading Activation Techniques in Information Retrieval. *Artificial Intelligence Review*, (11):453–482, 1997.
9. Alan J. Dix, Akrivi Katifori, Giorgos Lepouras, Costas Vassilakis, and Nadeem Shabir. Spreading activation over ontology-based resources: from personal context to web scale reasoning. *Int. J. Semantic Computing*, 4(1):59–102, 2010.
10. Martin Fowler. *Refactoring: Improving the Design of Existing Code.* Addison-Wesley, Boston, MA, USA, 1999.
11. Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns.* Addison-Wesley Professional, January 1995.

12. Qi Gao, Junwei Yan, and Min Liu. A Semantic Approach to Recommendation System Based on User Ontology and Spreading Activation Model. In *NPC '08: Proceedings of the 2008 IFIP International Conference on Network and Parallel Computing*, pages 488–492, Washington, DC, USA, 2008. IEEE Computer Society.
13. Jose Emilio Labra Gayo, Patricia Ordoñez de Pablos, and Juan M. Cueva Lovelle. Combining collaborative tagging and ontologies in image retrieval systems. 2007.
14. Jose Emilio Labra Gayo, Patricia Ordoñez de Pablos, and Juan Manuel Cueva Lovelle. Wesonet: Applying semantic web technologies and collaborative tagging to multimedia web information systems. *Comput. Hum. Behav.*, 26:205–209, March 2010.
15. Fatih Gelgi, Srinivas Vadrevu, and Hasan Davulcu. Improving Web Data Annotations with Spreading Activation. In *WISE*, pages 95–106, 2005.
16. Stephan Gouws, G-J van Rooyen, and Herman A. Engelbrecht. Measuring conceptual similarity by spreading activation over wikipedia's hyperlink structure. In *Proceedings of the 2nd Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources*, pages 46–54, Beijing, China, August 2010. Coling 2010 Organizing Committee.
17. Maurice Grinberg, Vladimir Haltakov, and Hristo Stefanov. Approximate spreading activation for efficient knowledge retrieval from large datasets. In *Proceeding of the 2011 conference on Neural Nets WIRN10: Proceedings of the 20th Italian Workshop on Neural Nets*, pages 326–333, Amsterdam, The Netherlands, The Netherlands, 2011. IOS Press.
18. Akrivi Katifori, Costas Vassilakis, and Alan J. Dix. Ontologies and the brain: Using spreading activation through ontologies to support personal interaction. *Cognitive Systems Research*, 11(1):25–41, 2010.
19. W. Liu, A. Weichselbraun, A. Scharl, and E. Chang. Semi-automatic ontology extension using spreading activation. *Universal Knowledge Management*, 0(1):50–58, 2005.
20. Scott Everett Preece. *A Spreading Activation Network Model for Information Retrieval*. PhD thesis, University Illinois, Urbana, IL, USA., 1981.
21. Yonggang Qiu and Hans-Peter Frei. Concept-based query expansion. In *Proceedings of SIGIR-93, 16th ACM International Conference on Research and Development in Information Retrieval*, pages 160–169, Pittsburgh, US, 1993.
22. Cristiano Rocha, Daniel Schwabe, and Marcus Poggi de Aragão. A Hybrid Approach for Searching in the Semantic Web. In *WWW*, pages 374–383, 2004.
23. Kinga Schumacher, Michael Sintek, and Leo Sauermann. Combining metadata and document search with spreading activation for semantic desktop search. In Sean Bechhofer, Manfred Hauswirth, Jörg Hoffmann, and Manolis Koubarakis, editors, *Proc. of ESWC*, pages 569–583. Springer, June 2008.
24. Ján Suchal. On finding power method in spreading activation search. In Viliam Geffert, Juhani Karhumäki, Alberto Bertoni, Bart Preneel, Pavol Návrat, and Mária Bieliková, editors, *SOFSEM (2)*, pages 124–130. Safarik University, Kosice, Slovakia, 2008.
25. Alexander Troussov, Mikhail Sogrin, John Judge, and Dmitri Botvich. Mining socio-semantic networks using spreading activation technique. 2008.
26. George Tsatsaronis, Michalis Vazirgiannis, and Ion Androutsopoulos. Word sense disambiguation with spreading activation networks generated from thesauri. In *Proceedings of the 20th international joint conference on Artifical intelligence*, pages 1725–1730, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

27. By Michael Wolverton, Michael Wolverton, Michael Wolverton, Barbara Hayes-roth, and Barbara Hayes-roth. Retrieving semantically distant analogies with knowledge-directed spreading activation. In *In Proceedings AAAI-94*, pages 56–61. AAAI Press, 1994.
28. Gui-Rong Xue, Shen Huang, Yong Yu, Hua-Jun Zeng, Zheng Chen, and Wei-Ying Ma. Optimizing Web Search Using Spreading Activation on the Clickthrough Data. In *WISE*, pages 409–414, 2004.