Akıllı Sistemlerde Yenilikler ve Uygulamaları Sempozyumu - 2012

SPARQL Sorgu Eniyilemesi için Karınca Kolonisi Yöntemi

4 Temmuz 2012

E. Güzel Kalaycı[1], T. E. Kalaycı[2]

1. Bilgisayar Mühendisliği, İzmir Ekonomi Üniversitesi, İzmir
2. Bilgisayar Mühendisliği, Celal Bayar Üniversitesi, Manisa

NE YAPIYORUZ?

- Bellekteki ontolojilerin sorgulanmasında kullanılan SPARQL sorgularının çalışma zamanını iyileştirmek için bir Karınca Kolonisi Eniyilemesi yaklaşımı sunuyoruz.
  - Temel çizge desenindeki (basic graph pattern) üçlü desenlerinin (triple pattern) sıralarını iyileştiriyoruz.
  - Önceden hesaplanmış veriler kullanmadan gerçek zamanlı iyileştirme yapıyoruz.

İÇİNDEKİLER

SPARQL

- SPARQL RDF verilerini sorgulamak için tanımlanmış sorgulama dilidir.
- Örnek Sorgu :

```
PREFIX c: <http://www.daml.org/2001/09/countries/fips#>
PREFIX o: <http://www.daml.org/2003/09/factbook/factbook-ont#>
SELECT ?partner ?neighbour WHERE
{  c:TU o:border ?tuBorder.
   ?tuBorder o:country ?border.
   ?border o:importsCommodity ?iCommodity.
   ?border o:industry ?industry.
   ?border o:importPartner ?impPartner.
   ?impPartner o:country ?iPartner. }
```

TEMEL ÇİZGE DESEN

- Temel çizge desen bir sorgunun üçlü desenlerinin kümesidir.

  c:TU o:border ?tuBorder.

  ?tuBorder o:country ?border.

  ?border o:importsCommodity ?iCommodity.

  ?border o:industry ?industry.

  ?border o:importPartner ?impPartner.

  ?impPartner o:country ?iPartner.

SPARQL Sorgu Eniyilemesi

- Üçlü desenlerinin yeniden sıralanması etkili bir SPARQL sorgu eniyilemesi yöntemidir.
- Üçlü desenlerini yeniden sıralamanın amacı sorgunun çalışma zamanını düşürmektir.

SPARQL Sorgu Eniyilemesi - Örnek

- Türkiye'nin komşularını ve bu komşuların ithal ettiği malları, sanayi kollarını ve ithalat partnerlerini sorgulayan sorgunun temel çizge deseni ***a***'da sıralanmıştır.
- *a*'nın çalışma zamanı 762 ms'dir.
- *b*'nin çalışma zamanı 163 ms'dir.

SPARQL Sorgu Eniyilemesi - Örnek

a) Eniyileme Öncesi

1. ?border o:importsCommodity ? iCommodity.
2. ?border o:industry ?industry.
3. c:TU o:border ?tuBorder.
4. ?tuBorder o:country ?border.
5. ?border o:importPartner ?impPartner.
6. ?impPartner o:country ?iPartner.

b) Eniyileme Sonrası

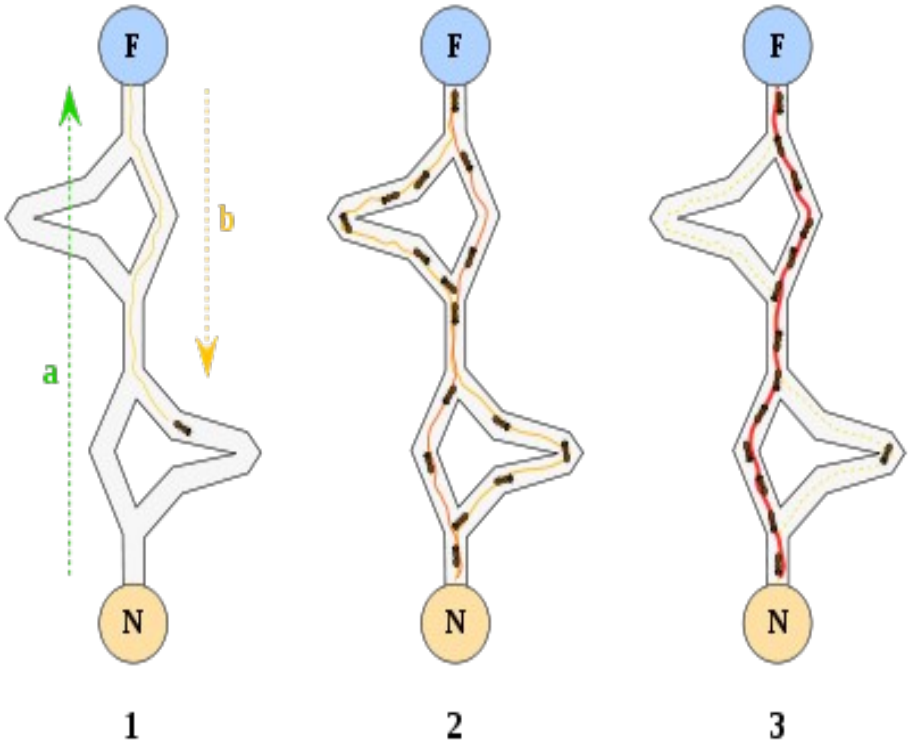
1. c:TU o:border ?tuBorder.
2. ?tuBorder o:country ?border.
3. ?border o:importsCommodity ? iCommodity.
4. ?border o:industry ?industry.
5. ?border o:importPartner ?impPartner.
6. ?impPartner o:country ?iPartner.

SPARQL Sorgu Eniyilemesi

- Üçlü desenlerinin sıralanmasındaki amaç birleştirme (join) maliyetlerinin (ara sonuç kümelerinin boyutlarının) minimize edilmesidir.

KARINCA KOLONİSİ ENİYİLEMESİ

- Karınca Kolonisi Eniyilemesi kombinatoryal eniyileme problemlerinin çözümü için kullanılan meta-heuristictir.
- Karıncaların yiyecek bulma davranışlarından esinlenilerek oluşturulmuştur.



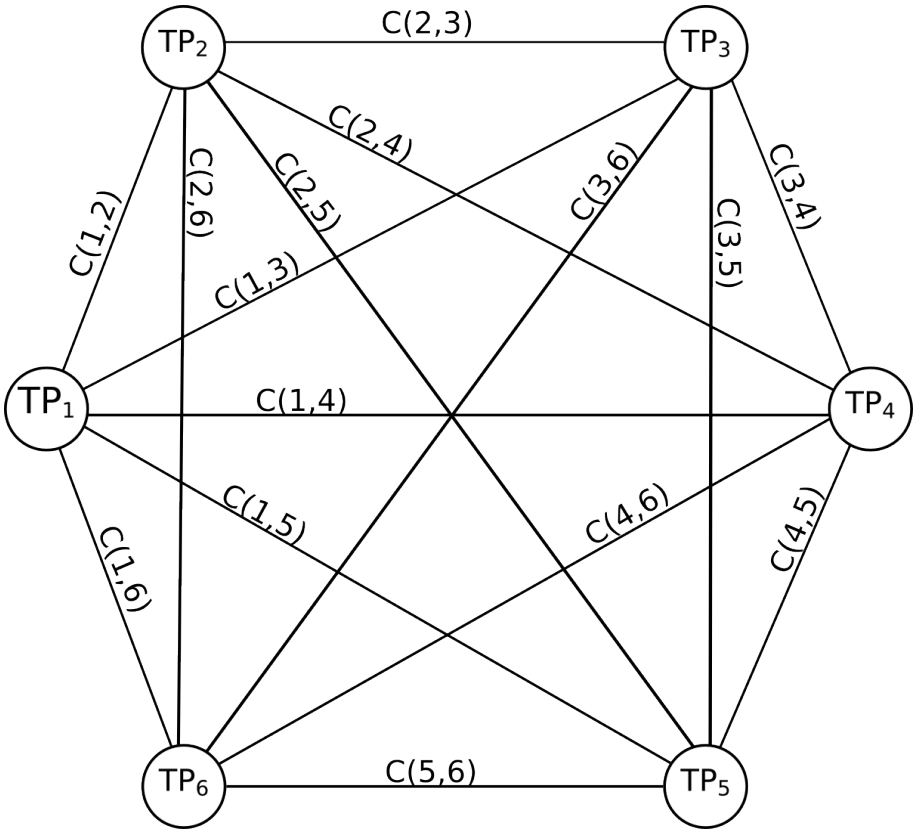
Resim Kaynağı: http://en.wikipedia.org/wiki/Ant colony optimization algorithms

KARINCA KOLONİSİ ENİYİLEMESİ

- Problemin çözümünde KKE algoritmalarından Karınca Sistemi algoritması kullanılmıştır.
    - Parametreleri belirle ve karıncaları ilkle
    - Sayaç tur sayısına eşit oluncaya kadar devam et
        - Düğümlerin olasılıklarını hesapla
        - Olasılığı en yüksek olan düğümü seç
        - Her karınca için tur uzunluğunu hesapla ve en iyi karıncayı seç
        - Buharlaşmayı gerçekleştir
        - Karıncanın maliyetini hesapla, feromon depola, karıncaları resetle
    - En iyi sonucu döndür

SOYUTLAMA

- Temel Üçlü Deseni tam çizge şeklinde soyutlanmıştır.
- Her düğüm bir üçlü deseni ve her kenar bağlı olduğu düğümlerin tahmini birleştirme maliyetlerini simgelemektedir.
- Bu çizge minimum maliyete sahip üçlü desen sırasını bulmaya çalışan karınca sistemi algoritmasına verilen girdiyi oluşturmaktadır.



MALİYET HESAPLAMA

- Birleştirme maliyetini hesaplamak için üçlü desenlerinin seçiciliğinden (üçlü deseniyle eşleşen üçlü (triple) sayısı / ontolojideki toplam üçlü sayısı) yararlanılmaktadır.
- Maliyet hesabı (kenar ağırlığı) iki adımda yapılmaktadır.
    1. Üçlü desenlerinin seçiciliğinin tahminlenmesi
    2. Birleştirme işleminin maliyetinin hesaplanması

SEÇİCİLİK TAHMİNİ

1. Değişken Sayımı (Variable Counting)
2. GSH (Graph Statistics Handler)

DEĞİŞKEN SAYIMI

- Üçlü desenlerinin bileşenlerinin

sel(Subject) < sel(Object) < sel(Predicate)

şeklinde derecelendirilmesine ve bağımlı (bound) ya da bağımsız (unbound) olarak sınıflandırılmasına dayanır.

GSH

- GSH en kesin tahminlemeleri yapan Jena tarafından sağlanan yöntemdir.
- Fakat birden fazla bağımlı bileşen içeren üçlü desenleri için tahminlemeyi desteklememektedir.

BİRLEŞTİRME İŞLEMİNİN MALİYET HESABI

1. Basit Maliyet
2. Birleştirme Maliyeti İçin Değişken Sayımı
3. Değiştirilmiş Değişken Sayımı

BASİT MALİYET

1. c:TU o:exportPartner ? expPartner .
2. ?expPartner o:country ?part.
3. ?part o:border ?bord.
4. ?bord o:country ?neighbour.
5. ?neighbour c:name ?name.

Yandaki temel çizge desen örneğinin sorgu yolu sırasını yanda olduğu gibi kabul edersek (1, 2, 3, 4, 5 şeklinde), bu sorgunun maliyeti

C = (|c1|x|c2|)+(|c2|x|c3|)+

(|c3|x|c4|)+(|c4|x|c5|)

şeklinde hesaplanır.

BİRLEŞTİRME MALİYETİ İÇİN DEĞİŞKEN SAYIMI

- Birleştirme tiplerinin (Subject-Subject, Subject-Object ...vb) derecelendirilmesine dayanır.
- Örn: 1. c:TU o:exportPartner ?expPartner .

  2. ?expPartner o:country ?part.

(1-2 → Object-Subject Join)

DEĞİŞTİRİLMİŞ DEĞİŞKEN SAYIMI

- Zİncİr ve zincir-yıldız sorgularının eniyileştirimesi gereksinimlerini karşılamak için Stocker ve diğerlerinin (2008) tanımladığı derecelendirme modifiye edilmiştir.
- Object-Subject birleştirmesinin seçiciliği 2 katına çıkarılmıştır.

GERÇEKLEŞTİRİM

1. Maliyet Hesapları
2. Karınca Sistemi Algoritması

MALİYET HESAPLARI

1. GSH + VC (Değişken Sayımı)
2. GSH + VC – M (Değiştirilmiş Değişken Sayımı)

KARINCA SİSTEMİ

- Karıncalar rastgele seçilmiş düğümlere yerleştirilir.

Transition formula

$$p_{ij}^{k} = \frac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l \in N_i^k}[\tau_{il}]^{\alpha}[\eta_{il}]^{\beta}} \qquad if \quad j \in N_i^k$$

Pheromone deposition formula

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \triangle\tau_{ij}^{k} \qquad \forall (i,j) \in L$$

Value of deposited pheromone formula

$$\triangle\tau_{ij}^{k} = \begin{cases} 1/C^{k}, if\ edge(i,j)\ belongs\ to\ T^{k}; \\ 0,\ otherwise; \end{cases}$$

(Kaynak: Guzel Kalayci & Kalayci, 2012)

DENEY ORTAMI

- Uygulama Java programlama dili kullanılarak Apache Jena Framework, ARQ sorgu motoru ortamında yapıldı.
- Deneylerde 95842 üçlü içeren CIA World Factbook ontolojisi kullanıldı.
- Her sorgu 10 defa çalıştırıldı ve elde edilen çalıştırma zamanlarının ortalaması alındı.

BASİT MALİYET DENEYLERİ

- Deneylerde 2, 4, 6, 8, 10 üçlü içeren zincir, yıldız ve zincir yıldız sorgular kullanıldı.

Tablo 1. İki üçlü desen içeren sorgular

	Sorgu-1 (Z)	Sorgu-2 (Z)	Sorgu-3 (Z)	Sorgu-4 (Z)
N(%)	100	100	100	100
KS (%)	86.43	90.35	95.31	104.69

BASİT MALİYET DENEYLERİ

Tablo 2. Dört üçlü desen içeren sorgular

	Sorgu-1 (Z)	Sorgu-2 (Z)	Sorgu-3 (Z)	Sorgu-4 (Z)
N(%)	100	100	100	100
KS (%)	0.23	1.92	3.19	7.08

Tablo 3. Altı üçlü desen içeren sorgular

	Sorgu-1 (Z)	Sorgu-2 (Y)	Sorgu-3 (Y)	Sorgu-4 (Z)
N(%)	100	100	100	100
KS (%)	17.67	38.74	62.69	30.15

BASİT MALİYET DENEYLERİ

Tablo 4. Sekiz üçlü desen içeren sorgular

	Sorgu-1 (Z+Y)	Sorgu-2 (Z+Y)	Sorgu-3 (Z+Y)	Sorgu-4 (Z+Y)
N(%)	*>1 saat*	*>1 saat*	100	100
KS (%)	56.16s	2.45s	74.56	78.10

Tablo 5. On üçlü desen içeren sorgular

	Sorgu-1 (Z+Y)	Sorgu-2 (Z+Y)	Sorgu-3 (Z+Y)	Sorgu-4 (Z+Y)
N(%)	100	100	100	100
KS (%)	63.47	0.53	28.31	72.63

BİLEŞİK MALİYET DENEYLERİ

- Deneylerde 4, 6, 8, 12, 14 üçlü desen içeren zincir, yıldız ve zincir-yıldız sorgular kullanılmıştır.
- 4 farklı çalıştırma tipi vardır:
  - NE (Normal Execution) (Herhangi bir eniyileştirme yapılmadan)
  - STO-VC (Stocker ve diğerleri tarafından 2008'de sunulan algoritma + VC)
  - RE (Jena'da varolan eniyileme algoritması)
  - AS-VC (Karınca Sistemi + VC)
  - AS-VC-M (Karınca Sistemi + VC-M)

BİLEŞİK MALİYET DENEYLERİ

(a) Queries with 4 triple patterns

	NE	RE	STO-VC	AS-VC	AS-VC-M
Q1	13435	6490	88	67	**63**
Q2	2828	2369	101	97	**74**
Q3	676	65	69	313	**46**
Q4	11371	11252	150	480	**127**

(b) Queries with 6 triple patterns

	NE	RE	STO-VC	AS-VC	AS-VC-M
Q1	310	291	67	241	**56**
Q2	140	126	176	156	**116**
Q3	258	163	230	300	**212**
Q4	**13**	**13**	79	107	66

(c) Queries with 8 triple patterns

	NE	RE	STO-VC	AS-VC	AS-VC-M
Q1	51873	7609	292	325	**277**
Q2	862	862	200	242	**192**
Q3	794	795	73	4056	**62**
Q4	49804	40972	**209**	279	230

(d) Queries with 10 triple patterns

	NE	RE	STO-VC	AS-VC	AS-VC-M
Q1	4328	4251	3780	3760	**617**
Q2	126559	5289	4176	4277	**2241**
Q3	4657	4738	1824	4714	**1441**
Q4	313	293	**70**	188	137

(e) Queries with 12 triple patterns

	NE	RE	STO-VC	AS-VC	AS-VC-M
Q1	24796	26410	**158**	502	378
Q2	5968	85	**72**	400	97

(f) Queries with 14 triple patterns

	NE	RE	STO-VC	AS-VC	AS-VC-M
Q1	19307	19219	2986	**1441**	2685
Q2	148241	153994	**216**	79389	247

(Kaynak: Guzel Kalayci & Kalayci, 2012)

SONUÇ

- Önerilen yöntem:
  - Bellekte tutulan ontolojilerde daha kısa çalıştırma zamanı için SPARQL sorgularının üçlü desen sıralarını iyileştirir.
  - Önceden hesaplanmış verilere ihtiyaç duymaz.
  - Çalışma zamanını normal çalışma zamanına göre önemli ölçüde düşürür.

DEVAM EDEN ÇALIŞMALAR

- Seçicilik tahmini ve maliyet hesabı için kullanıan sezgileri her türlü çizge yapısına sahip sorguların çalışma zamanını düşürmek amacıyla iyileştirmek
- Deneyleri farklı sorgu ve ontolojiler kullanarak çalıştırmak
- Problem için farklı eniyileme algoritmaları uygulamak
- KKE'ye yerel arama algoritmaları eklemek

KAYNAKLAR

- E. Guzel Kalayci, T. E. Kalayci, “Reordering Triple Patterns of SPARQL Queries using Ant Colony Optimization”, Proc. of 18th Int. Conf. on Soft Computing (MENDEL 2012), Brno, Czech Republic, 27-29 June, 2012, ISBN: 978-80-214-4540-6
- JENA - http://incubator.apache.org/jena/
- Java - http://www.oracle.com/tr/technologies/java/
- http://www.cia.gov/library/publications/the-worldfactbook/index.html
- http://sourceforge.net/projects/jena/files/ARQ/ARQ-2.6.0/

KAYNAKLAR

- Abdel Kader, R. and van Keulen, M. Overview of query optimization in xml database systems. Technical Report TR-CTI, EEMCS, University of Twente, Enschede, 2007.
- Berners-Lee, T., Hendler, J., and Lassila, O. The semantic web. Sci. Am., 284(5):34–43, 2001.
- Dorigo, M. and Stützle, T. Ant Colony Optimization. MIT Press, Cambridge, MA, 2004.
- Harris, S. and Seaborne, A. SPARQL 1.1 Query Language - W3C Working Draft 05 Jan. 2012. 2012.
- Hartig, O. and Heese, R. The sparql query graph model for query optimization. In Proc. of the 4th European Conf. On The Semantic Web: Research and Applications, ESWC 07, pages 564–578, 2007.

KAYNAKLAR

- Hogenboom, A., Milea, V., Frasincar, F., and Kaymak, U. Rcq-ga: Rdf chain query optimization using genetic algorithms. In Proc. of the 10th Int. Conf. on EC-Web, pages 181–192, 2009.
- Hogenboom, F., Hogenboom, A., van Gelder, R., Milea, V., Frasincar, F., and Kaymak, U. Qmap: An RDF-based queryable world map. In 3rd Int. KMO Conf., pages 99–110, 2008.
- Ioannidis, Y. E. Query optimization. ACM Comput. Surv., 28(1):121–123, 1996.

KAYNAKLAR

- Maduko, A., Anyanwu, K., Sheth, A., and Schliekelman, P. Estimating the cardinality of rdf graph patterns. In Proc. Of the 16th Int. Conf. on World Wide Web, pages 1233–1234. ACM, 2007.
- Maniezzo V, Gambardella L.M., D. L. F. New Optimization Techniques in Engineering, chapter Ant Colony Optimization, pages 101–117. Springer-Verlag, 2004.
- Neumann, T. and Weikum, G. Rdf-3x: a risc-style engine for rdf. Proc. VLDB Endow., 1(1):647–659, 2008.

KAYNAKLAR

- Ozsu, M. T. and Blakeley, J. A. Query processing in object-oriented database systems. In Modern Database Systems, pages 146– 174. ACM Press and Addison-Wesley, 1995.
- Ruckhaus, E., Ruiz, E., and Vidal, M. Query evaluation and optimization in the semantic web. Theory Pract. Log. Program., 8(3):393–409, 2008.
- Shironoshita, E. P., Ryan, M. T., and Kabuka, M. R. Cardinality estimation for the optimization of queries on ontologies. SIGMOD Rec., 36(2):13–18, 2007.

KAYNAKLAR

- Stocker, M., Seaborne, A., Bernstein, A., Kiefer, C., and Reynolds, D. Sparql basic graph pattern optimization using selectivity estimation. In Proc. of the 17th Int. Conf. On WWW, pages 595–604. ACM, 2008.
- Stuckenschmidt, H., Vdovjak, R., Broekstra, J., and Houben, G. Towards distributed processing of rdf path queries. Int. J. Web Eng. Technol., 2(2/3):207–230, 2005.