Artif Intell Rev (2008) 29:287–303
DOI 10.1007/s10462-009-9141-6

On the importance of relational concept knowledge in referral networks

Tony White · Shaun McQuaker · Amirali Salehi-Abari

Published online: 1 November 2009

Abstract This paper deals with the topic of peer-to-peer referral systems and the policies that allow for the emergence of efficient retrieval of requested information. In an agent-based peer-to-peer network, member agents are capable of giving and following referrals to each other. This results in the emergence of communities where agents directly interact with other neighboring agents that supply the required service or will refer the right source. The notion of referral networks, as presented in the work of (Yolum and Singh, Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, pp 592–599, 2003), and their application to knowledge management, lacks two fundamental aspects; the relation of concepts within a domain, and the ability of an agent to dynamically change their interests based on suggestions in the form of concept relations. This paper introduces the concept of an oracle agent, which is an agent with relational concept knowledge that can supply suggestions to a querying agent on how to adapt their interests. Additionally the notion of health and localized trust automata are used to aid agents in discriminating useful pieces of concept knowledge. These new features allow agents to search in new ways so as to achieve superior results and as a consequence outperform agents in a traditional referral network. The paper presents simulation results that confirm this hypothesis.

Keywords Multi-agent system · Referral networks · Relational concept knowledge · Oracle

T. White (✉) · S. McQuaker · A. Salehi-Abari
Carleton University, 1125 Colonel By Drive, Ottawa, ON, Canada
e-mail: arpwhite@scs.carleton.ca

S. McQuaker
e-mail: shaun.mcquaker@gmail.com

A. Salehi-Abari
e-mail: asabari@scs.carleton.ca

1 Introduction

Information retrieval researchers have suggested that the process of seeking information has always been a social process (Page et al. 1998), thus applying social networking concepts to internet searching has the potential to alleviate some of the complexities of online searching by forming communities around sources of expertise.

In the real world, humans innately relate concepts. It is our ability to relate concepts and form hierarchies of relations that allows us to effectively process and generalize data. This concept relation knowledge is of great importance; it allows us to change our interests to reflect new found knowledge, rephrase questions in a manner that can return better search results, and give suggestions to people asking us questions, in the form, "Did you mean this instead of that?". Learning that a topic is related to another and then learning about this new topic in the long run educates you on the original topic. It is this observation that motivates the research described in this paper.

Current work for modelling social networks with referral networks does not provide two essential ingredients that humans use in real-life social networks: concept knowledge and suggestions which *propagate* concept knowledge. The ability for referral networks to provide a mechanism for the dissemination of concept knowledge through suggestions and thus the ability to distinguish context is essential to reducing the complexity of searching through the vastness of knowledge available on the Internet today. This paper, therefore, proposes solutions to the concept knowledge propagation problem within the scope of referral networks.

This paper builds on top of the existing work in the field of Referral Networks by extending how knowledge is maintained and modelled in those networks. The objective is to study the effect of relational concept knowledge on the quality, efficiency and authoritativeness of a referral system. Thus, how agents within a referral network with concept knowledge assist other agents in finding useful knowledge via suggestions is examined and several learning algorithms are proposed for concept propagation, learning, and evaluation. Additionally, localized concept knowledge trust is modelled using a localized trust automaton to aid in minimizing risk of using useless or malicious pieces of concept knowledge. The notion of agent health is also introduced as a reinforcement technique to drive evaluation of concept knowledge.

This paper consists of three further sections. Section 2 provides an introduction to referral networks and provides the necessary machinery to extend them relational concept knowledge. Section 3 describes two experiments which demonstrate the utility of the additional knowledge. Section 4 concludes with a summary of key messages for the paper.

2 Referral networks

2.1 Referral networks: an overview

Expertise, interest and sociability within a referral network are represented using the Vector Space Model (Salton and McGill 1983). For knowledge representation in a domain, each dimension in an interest or expertise vector represents the amount of expertise or interest an agent has in that domain, normalized between 0.0 and 1.0, where 1.0 represents a high interest or expertise in a domain and 0.0 represents a low interest or expertise in a domain.

Several evaluation metrics are used in referral networks. These metrics are described in the following five sections.

2.1.1 Similarity

Similarity is computed as shown in Eq. 1 (Yolum and Singh 2004) below, where I_i represents the interest vector of agent i, I_j is the interest of another agent j and n is the dimension of I_i and I_j (they must be of the same length). Similarity is commutative, measures the Euclidean distance between two vectors and is normalized to obtain a result between 0.0 and 1.0 (Yolum and Singh 2003).

$$I_i \otimes I_j = \frac{e^{-\| I_i - I_j \|^2} - e^{-n}}{1 - e^{-n}} \tag{1}$$

2.1.2 Capability

Capability allows an agent to reply with a certain degree of surety or else give a referral. Capability resembles cosine similarity but also takes into account the magnitude of the expertise vector (Yolum and Singh 2003). Capability is computed as follows (Yolum and Singh 2002):

$$Q_i \oplus E_j = \frac{\sum_{t=1}^{n} q_t e_t}{\sqrt{n \sum_{t=1}^{n} q_t^2}} \tag{2}$$

In Eq. 2, Q_i is agent i's query vector $[q_1, q_2, \ldots, q_n]$ and E_j is the expertise vector $[e_1, e_2, \ldots, e_n]$ of agent j, again n is the dimension of Q_i and E_j.

2.1.3 Effectiveness

The effectiveness of a system measures how easily agents find useful providers (Yolum and Singh 2003). There are two metrics used to measure effectiveness. The first is *direct quality*, which is measured as the usefulness of the direct neighbours of an agent, in terms of their expertise and the agent's interests. The second is *n*th best quality. This metric takes into consideration "how well the agent's interest matches the expertise of all other agents in the system, scaled down with the number of agents it has to pass to get to the agent" (Yolum and Singh 2003). The value of n is taken to be twice the number of neighbours for a given agent as in (Yolum and Singh 2003). The contribution of agent j to agent i's quality is given by:

$$\frac{I_i \oplus E_j}{path(i, j)} \tag{3}$$

2.1.4 Efficiency

Efficiency is a measure of how well experts are responding to questions within the network. Accuracy is measured over all agents in the network and is calculated using Eq. 4. The number of correct answers is the sum of correct answers received by all agents; the total questions asked is the sum of all questions asked by agents.

$$Accuracy = \frac{Correct\ Answers}{Total\ Questions\ Asked} \tag{4}$$

Table 1 Example expert and consumer vectors

	Theory of relativity	Field equations	Newtonian physics
Expert	0.1	0.8	0.8
Consumer (before)	0.8	0.1	0.8
Consumer (after)	0.8	0.8	0.8

2.1.5 Authoritativeness

Some agents will be chosen as a neighbour by a greater number of other agents, and are thus identified as more authoritative. The authoritativeness measure is computed using the PageRank (Page et al. 1998) metric to study the emergence of authorities (Yolum and Singh 2003) in referral networks.

2.2 Relational concepts

Within the Vector Space Model (VSM), it is quite possible that the expertise vector of an expert and the query vector of a consumer are very different upon analysis using the similarity metric described in Eq. 1. However, it is quite plausible that the interests of the consumer lie in the same domain as the expertise vector of the expert and may even be tightly related. Thus a small amount of concept relational knowledge would be of great benefit, if the consumer were aware of it. This is the essence of the algorithms provided by this paper and the primary motivation for the addition of a new type of agent—the oracle.

As can be seen in Table 1, being aware of relational concept knowledge allows an agent to dynamically change its interests to better learn about concepts in the domain of its interests. The first row of Table 1 depicts the expertise vector of an expert in the domain of Field Equations, while the second depicts a consumer interested in the Theory of Relativity. If a consumer were to ask the expert about the Theory of Relativity it would not get an answer. However, after learning that Field equations and the Theory of Relativity are related, if the consumer were to modify its interests placing more of an onus on Field Equations and ask a question based on these new found interests, the expert would then be able to answer with a much better degree of accuracy. This would lead to an improved, presumably satisfactory, response and thus the querying agent would learn about Field Equations and thus the Theory of Relativity.

Notice that interest vector in the third row of Table 1 now matches the expertise vector in the first row in two of the three areas instead of one of the three areas as in the second row. This will result in the query (based on the interests of the querying agent) being more similar to the expertise vector of the expert and thus is more likely to overcome the capability threshold and be answered by the expert.

For a referral network to function as described above, concept relational knowledge needs to be introduced into the model along with policies for its usage. Concept knowledge is contained within oracle agents.

The oracle agent with concept knowledge now contains a set of concept relations described in Data Structure 1. Concept relations are modelled by two numbers representing the indexes within the VSM that are related. The concepts are unidirectional.

Data Structure 1: Concept-Relation

```
1: int domainA;
2: int domainB;
3: Agent suggestingAgent;
```

We now proceed by describing the modified referral network agent environment.

2.3 Simulation algorithms

There are two main algorithms employed in a referral network. The first is the algorithm to ask queries and the second is to answer queries. Agents ask and evaluate queries in accordance with the following algorithm as defined in (Yolum and Singh 2003). Lines 1 through 20 in the Algorithm 1 given below are repeated a set number of times for every cycle; the bold text representing modifications for concept relation processing. At the end of each cycle agents are allowed to change neighbours.

Algorithm 1: Ask-Query()

```
1:  Generate Query (query generation policy, Sect. 2.3.1)
2:  Send query to matching neighbours (asking policy, Sect. 2.3.2)
3:  While (!timeout) do
4:    Receive Message
5:    If message.type is a referral then
6:      Send query to referred agent
7:      Add referral to referral graph
8:    Else
9:      Add answer to answerset
10:   End if
11: End while
12: For i = 1 to |answerset| do
13:     if answer(i) is a suggestion then
14:       Update concept knowledge store
15:   end if
16:   Evaluate answer(i)
17:   Update agent models (learning policy, Sect. 2.3.3)
18:   End for
19: UpdateInterests(Interest Vector) (interest adaptation policy, Sect. 2.6.2)
20: ChangeNeighbours (neighbour selection policy, Sect. 2.3.6)
```

2.3.1 Query generation policy

Since there is no user interaction during a simulation, a policy for generating queries is required. The query generation policy fulfills this need by simply iterating over each index in the interest vector of the agent and perturbing the value by some small random amount from a uniform distribution. This newly perturbed interest vector is then used as the query.

2.3.2 Asking policy

The asking policy is responsible for determining if the neighbour agent should be asked the given query. The default implementation of this policy uses the current agent's models of all neighbours and determines if the neighbour is capable (using the capability metric) of answering for the current query. If the capability metric is above a pre-defined threshold then the neighbour is queried.

2.3.3 Learning policy

The first responsibility of the learning policy is to evaluate the answers received for a query. If the similarity of current query and an answer (as determined using the capability metric) is above a given threshold then the answer is graded as good otherwise it is graded bad. The best answer given to agent i is chosen using Eq. 5.

$$\forall r \in R \; ; \; B_i = \max(r \oplus Q_i) \tag{5}$$

where R is the set of all responses (answers) from agents who responded to the query Q_i from agent i, and B_i is the best answer given to agent i.

The learning policy then updates three major areas, the modelled expertise and sociability of the agents which have provided answers or referrals and the expertise of the agent asking the question. The modelled sociability vector of agent j is updated according to Algorithm 2 if it gives a referral that leads to a good answer.

Algorithm 2: Update Sociability of Good Answer

1: For k = 0 to size of S_j
2: $S_j[k] = S_j[k] + ((1 - S_j[k])^*\beta)/\tau_j$
3: End For

And if a bad or no answer is given the sociability of agent j is updated using Algorithm 3.

Algorithm 3: Update Sociability of Bad or No Answer

1: For k = 0 to size of S_j
2: $S_j[k] = S_j[k] - (S_j[k]^*\beta)/\tau_j$
3: End For

In both algorithms, β denotes a weighting coefficient, τ_j denotes the distance (as recorded in the referral graph for the query, line 7, Algorithm 1) between agent j and the agent who provide agent i a good, bad or no answer at all, and S_j represents the modelled sociability vector of agent j. The referral graph is used to ascertain τ_j and is a directed a-cyclical graph (DAG) rooted at agent i (which made the query), contains a finite set of agents (vertices) and a set of referrals (edges). The properties of the DAG are explained in (Yu 2001) and are not repeated here.

After updating the sociability of an answering agent, the modelled expertise of answering agents as well as the expertise of the querying agent is updated. The expertise of the agent i making the query updates the modeled expertise of agent j who gave an answer, using Algorithm 4. This algorithm is used for all agents that give an answer.

Algorithm 4: Update Modeled Expertise

1: For each agent j giving an answer
2: E_j = modelled expertise vector of agent j
3: R_j = answer vector given by agent j
4: For k = 0 to size of E_j
5: $E_j[k] = (1 - \alpha')^* E_j[k] + (\alpha'^* R_j[k])$
6: End For
7: End For

E_j denotes the modelled expertise vector of agent j giving the answer, and α' denotes a learning coefficient.

To update its own expertise (E_i) the querying agent uses the best answer given (B_i) and a different learning coefficient α. Algorithm 5 formalises the computation. The expertise value is bounded by a target value, T_i.

Algorithm 5: Update Self Expertise

1: For k = 0 to size of E_i
2: $E_i[k] = min(T_i[k], (1 - \alpha) * E_i[k] + (\alpha * B_i[k]))$
3: End For

During each cycle an agent will invariably receive queries which it will attempt to answer by using Algorithm 6 from (Yolum and Singh 2003); the highlighted text representing modifications for concept relation processing. In Algorithm 6, the agent receiving the query is j and the agent posing the query is i, E_j denotes the expertise of agent j, and Q_i denotes the query posed by agent i.

Algorithm 6: Answer-Query(Query Qi)

1: **If hasAGoodSuggestion(Q_i) *returns a concept relation then***
2: **Send concept relation to agent I (suggestion policy, Sect. 2.6.1)**
3: **End if**
4: If $E_j \oplus Q_i$ >CAPABILITY_THRESHOLD_FOR_ANSWERING_QUESTIONER then
5: Generate Answer for agent i
6: Else
7: Refer Neighbours for agent i
8: End if

2.3.4 Answering policy

The capability metric (Eq. 2) is used in line 4 of Algorithm 6 to determine if the agent has sufficient expertise to answer the query. The answer is then generated by perturbing the expertise vector by either some small amount or not at all (the latter was done in this paper).

2.3.5 Referral policy

The capability metric is also used when determining which neighbours to refer (line 7) of Algorithm 6. The query is compared to the modelled expertise of the neighbouring agents

and if sufficiently similar the agent is referred. There are three main referral policies *Refer All*,*Refer All Matching*, and *Refer Best*. The Refer All policy refers all neighbours to a querying agent, regardless of the neighbour's ability to answer the current query. Refer All Matching only refers neighbours whose modelled expertise capability exceeds some threshold. Refer Best will only refer the neighbour best suited to answer the query; i.e., the best expertise.

2.3.6 Neighbour selection policy

Once an agent has completed asking and answering a set of questions, they are allowed to choose new neighbours based on their updated agent models (as determined in line 20 of Algorithm 1). An agent must always keep a static number of neighbours; thus if an agent decides to choose a new neighbour it must drop an existing one. The neighbour selection policy utilizes Algorithm 7 to change neighbours.

Algorithm 7: Change-Neighbours()

```
1:  For all non-neighbouring agents
2:    Calculate Relevance of agent
3:    If relevance better than worst neighbour
4:      Replace neighbour with agent
5:    End If
6:  End For
```

Let i denote the agent changing neighbours and j one of its non-neighbours. The relevance of agent j (line 2) to agent i is calculated according to Eq. 6.

$$Rel_i\,(j) \;=\; (1-W)\;(I_i \;\oplus\; E_j) \;+\; W\;(I_i \;\oplus\; S_j) \tag{6}$$

where W is a coefficient weighting the importance of expertise or sociability in the calculation, I_i indicates the interest of agent i, S_j and E_j denote the modelled sociability and expertise respectively of the non-neighbouring agent j.

2.4 Health

The health of an agent represents how well it is learning about required domains of knowledge. If an agent is adapting its interests in a manner which will lead to the expertise that it requires, then its health will benefit, if not its health will suffer. The purpose of health in a referral network is twofold: Perpetuate a reinforcement learning cycle which causes the agent to eventually adapt interests in a useful way and provide the basis for a utility metric which drives trust in other agents.

The health of an agent is modelled as the absolute difference between its current expertise vector and some target expertise vector, which is given by the human operator at the start of the simulation. The target expertise vector is essentially a "yard-stick" measure of how well an agent is learning intended concepts. As the agent learns more related concepts the health of the agent will move towards zero, indicating that it has learned all needed concepts. The equation of measuring health is given below.

$$Health(i) = \sum_{i=0}^{z} |G_i - E_i| \tag{7}$$

In Eq. 7, the target expertise for agent i is represented by G_i, the current expertise of the agent is E_i, and z is size of the expertise vector E_i. A health value of zero denotes perfect health, i.e., it has learned all related concepts. Any value between zero and the size of the expertise vector denotes an increasingly worse health rating. Health measurements are read after each query and are stored for historical analysis and used in Algorithm 8.

2.4.1 The need for health

While quality measures how easily agents find useful providers it does not show how well agents are learning domains that they need to learn. An agent may have poor health yet the quality of the network is good, this situation would arise if the agent is, for instance, using a concept relation which is useless to the agent, but for which there are many experts in the network. Conversely an agent may have good health but the quality of the network may be poor. For instance, an agent may have just finished using a useful piece of concept knowledge and has learned about a needed domain, but now has moved onto a piece of concept knowledge which suggests a domain for which there are few or no experts in the network. As these examples illustrate the quality, effectiveness and authoritativeness of a network are irrelevant if agents are not actually learning about useful domains. Thus health is a measure to evaluate how well an agent is learning and is essential to evaluate the utility of concept relations.

2.4.2 Health stagnation

Health stagnation serves as an indicator that an agent has completed learning in a given domain. It is assumed that the agent's health will never degenerate in the lifecycle of a simulation. A method is therefore needed to determine if the health of the agent has reached a plateau and is not significantly improving. The number of health readings compared is determined by the configuration option H_r (the maximum number of health readings to determine health stagnation). The health is said to have stagnated if the difference of the summation of consecutive health readings is within some threshold H_t. Health stagnation is determined by using Algorithm 8.

Algorithm 8: Health Stagnation

1: X = Maximum health readings to determine stagnation (H_r)
2: Move back X+1 positions in the health list
3: Loop from current position to end of health list
4: diff += currentHealthReading – nextHealthReading
5: End Loop
6: Return abs(diff) < H_t

The H_r variable is the determining factor for how quickly concepts are used. The smaller the value of H_r the quicker concepts will be used. There is a trade-off, however, if H_r is too small agents may not have enough time to find experts in the network and potentially useful concept knowledge will be rendered useless; additionally the agent that suggested the concept knowledge will be treated as untrustworthy, when, in fact, this may not be the case. If H_r is too big agents will most likely find useful experts, but may take longer than necessary to realize that they have learned sufficiently. Throughout the simulations in this

paper, H_r is tuned to a value of 5 to ensure that agents have sufficient time to adapt their interests sufficiently to find experts in the system.

The H_t simulation variable also affects how concepts are used. A large value of H_t and large variations in health will be considered stagnant, which is not desirable. Thus, a small value for H_t was used (0.1) in this paper to ensure that, over 5 (H_r) successive readings the agent had learned sufficiently in a related domain.

2.5 Trust

Trust can be thought of as the qualified reliance on received information. Implicit in this description is that trust must be evaluated locally in order for it to be reliable; it must provide some utility or else it cannot be relied upon.

Local evaluation of answers in a referral network is achieved by evaluating the effect it has on the agent; i.e., its health. The measure of health can then be used to locally evaluate concept knowledge the agent has used. There is no distinction between malicious and useless concept relations; if the health of the agent gets better by using a concept relation it can then place more trust in the oracle who gave the suggestion, alternatively if the concept provided no utility less trust is placed in the oracle. Trust can then be used to determine future usage of concepts from the oracle.

2.5.1 Trust functions

Degrees of trust are calculated using trust functions, f. Trust functions are used within an agent to model the amount of trust placed in other oracles. A sigmoid function returning values in (0,1) was used. Trust is ultimately represented by associating an automaton with a trust function. Each agent contains a trust automaton for each oracle which provides it with a concept relation. Thus the level of trust of agent i in oracle j is defined as $TA(x)_{i,j}$, where x is the state of the trust automaton TA, and $TA(x)_{i,j} = f(x)$, where f is the trust function associated with TA. Increasing the automaton state will decrease trust in an oracle, while decreasing the automaton state will increase trust in the oracle. Agents are initially naïve which means agent i will assign the trust automaton TA for agent j to state 0 ($TA(0)_{i,j}$) upon receiving the first concept relation from agent j. The state 0 is a special case and always returns the value 0, that is $f(0) = 0$.

2.6 Suggestion and interest adaptation policy

For a referral network to function as described above, two new policies need to be introduced. One is a suggestion policy and deals with what concept knowledge an agent will suggest to another agent upon receiving a query. The second is an interest adaptation policy and functions in the capacity of updating the interests of an agent that has concept knowledge about related topics.

2.6.1 Suggestion policy

Within the referral network simulator the suggestion policy is used when an agent is asked a question. To determine a good suggestion for a given query an agent must determine the best concept relation in accordance with Algorithm 9. The policy outlined below is a stateful suggestion policy that attempts to aid the querying agent in the domain of knowledge for

which it has the most interest. Thus this policy is optimistic, representing a "best bang for the buck" implementation, and works on the assumption that where there is the most amount of interest, learning a concept relation for that domain will make the most difference in the quality and effectiveness of subsequent queries. To reduce the exchange of useless information an agent will never send the same agent the same piece of concept knowledge more than once. The suggestion policy used to determine the best suggestion to offer is detailed in Algorithm 9.

Algorithm 9: hasAGoodSuggestion(Query Q_i)

```
1:  largestIndex = index in Q_i with largest value
2:  Loop through all concept relations in set
3:    conceptRel = current concept
4:    indexA= Q_i .get(conceptRel.conceptA);
5:    If indexA is largest index in Q_i
6:      If haven't sent concept to querying agent
7:        Return conceptRel;
8:      End If
9:    End If
10: End Loop
11: Return bestConceptRel;
```

2.6.2 Interest adaptation policy

The core mechanism by which agents learn about related domains lies in the interest adaptation policy, shown in Algorithm 10. It is responsible for adapting interests using concept knowledge and determining if the concept used was helpful to the agent. If the concept is useful then trust in the agent that provided the concept knowledge is increased otherwise it is decreased.

Algorithm 10: Interest Adaptation

```
1:  If health isn't perfect and has stagnated (Algorithm 8)
2:  If no concept is selected then select next available concept
3:  If health has been stagnant for S_c cycles
4:    If health is better then increase trust in agent which gave concept knowledge
5:    If health is same then decrease trust in agent which gave concept knowledge
6:    For each concept C in unused concepts
7:      If have or ever had interest in domainA of C
8:        If trust in suggesting agent of C is higher than random number between 0 and 1
9:        curConcept = concept C
10         Reduce other areas of interest and exit For loop
11:     End If
12:     End For
13:   End If
14:   End If
15: If health is perfect or have used all concepts then exit
16: k = domain B of curConcept
17: I_i[k] = min(I_o[k], ξr)
```

Health stagnation is used to determine if an agent needs to use a concept relation. If querying with the current set of interests has caused the agent's health to stagnate (line 1) the agent attempts to use the next available concept (line 2). However, if health has stagnated and the agent is currently using a concept relation it will attempt to use it for S_c cycles before giving up. This is necessary as health stagnation motivates two different aspects of this algorithm; when to choose a new concept relation and when to stop using the current one. The S_c variable was chosen as 5 for simulations in this paper as it took no more than 5 cycles for consumer agents to find experts in the network. It is the realization of the author that this method is rather simplistic and may not be optimal in diverse simulation environments; however, the goal of this paper is not to provide the *best* way to determine when to stop using a concept relation, but rather a *simple* and *useable* method.

Lines 4 and 5 use a trust automaton for each agent giving concept relations to represent trust. Agents will decrease the state of the automaton in line 4 (increase trust) if health is benefited and increase the state of the automaton in line 5 (increase distrust) if health is not improved. Additionally the trust automaton is used in line 8 to read the current value of the trust function as determined by the state of the automaton, and determine if an agent is trustworthy enough to use their suggested concept relation.

The interest of agent i is increased in line 17, where I_i, denotes the interest vector of agent i; r is some random real number between 0 and 1, ξ denotes an interest increment coefficient and was set to 0.5 for all simulations, and I_o denotes the original interest vector of agent i before any interest adaptation using the current concept. Note that this calculation does not allow interest in the related domain to become larger than the amount of interest shown in the original domain. It is assumed that an agent will never want to show more interest in a related domain than in the original domain.

3 Experimentation

Three classes of experimentation were performed. The first demonstrates how referral networks with concept relations provided by oracles have efficiency properties comparable to networks without them. The results are not included here owning to space constraints. The second class shows that relations that depend upon other relations can be learned by the system—the transitive problem. Finally, a context experiment is performed where two oracles each have a concept relation relating a single concept (say 0) to another (either 1 or 2). The goal of this last class is to have agents searching for different things learn which oracle should be trusted.

3.1 Transitivity

It is nearly impossible to learn about one thing in isolation. Take, for instance, the real-world situation of learning about Einstein's general theory of relativity. It is nearly impossible to learn about Einstein's general theory of relativity without becoming aware of space-time curvature, which in turn introduces black holes, event horizons, worm-holes and so on. Thus learning about the general theory of relativity leads to transitively learning about other related domains. The objective of this simulation is to demonstrate agents can learn transitive concept relations.

This simulation again splits the experts into two groups one with expertise in domain 1 and the other with expertise in domain 2. One oracle contains two pieces of transitive concept

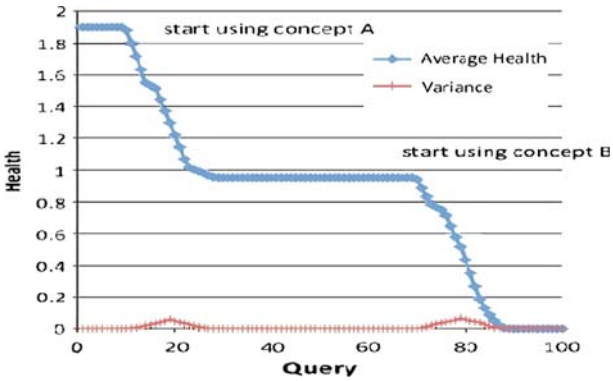

Fig. 1 Average health and variance

knowledge; the first relates domain 0 to 1 (concept A), the second relates 1 to 2 (concept B). Consumer agents are initially interested in domain 0 but would benefit from learning from experts in domain 1 and 2 (related domains). Thus they benefit initially from utilizing the concept relation $0 \rightarrow 1$. As a consequence of showing interest in domain 1 agents are then provided the concept relation $1 \rightarrow 2$ from the oracle, which in turn provides utility to the consumer agents.

3.1.1 Health results

The health metric demonstrates the effect of learning transitive concept relations, as it shows implicitly several features. First, concepts are propagated from the oracle to the consumer agents. If related concepts were not propagated the health of agents would never improve, as they would never learn about needed domains. Second, Experts are found for both domains. Without finding experts agents could never learn and thus their health would suffer.

Figure 1 depicts the average health and the variance in the health metrics of agents after each query. Consumer agents begin using concept A at query 10, hence the small plateau at the start of the curve. The descending curve to query number 30 denotes the time when agents are learning about domain 1. Also during this time the oracle provides concept B ($1 \rightarrow 2$). The second plateau from cycle 30 to 70 indicates that agents have completely learned about domain 1 using concept A. Once their health has stagnated for enough time, consumer agents then begin using concept B at cycle 71 and continue to learn about domain 2 until cycle 91, where they exhibit perfect health. Some agents may find experts quicker than others and begin learning earlier than others and the variance of the average health increases. Once all consumer agents have learned the concept the variance settles to zero. This is shown for concept A, where agents start learning about domain 1 at cycle 10; some learn earlier and thus the variance in health increases, but then begins to subside at cycle 20 as all agents learn domain 1. This is shown for concept B as well at cycle 71, where variance increases and then begins to subside around cycle 81, and eventually settles to 0 at cycle 92.

3.1.2 Quality

A powerful aspect of this transitivity simulation is its ability to demonstrate that a referral network with concept knowledge can rival the quality of a referral network without concept knowledge where the "right set of questions to ask" (interests) is known initially. For

Table 2 Comparison of quality

Measure	RF1	CRF concept A	RF2	CRF concept B
Direct quality	0.4311	0.4636	0.4311	0.4201
Nth best quality	0.2152	0.2315	0.2152	0.2049

Table 3 Concept propagation

Query number	Concepts shared	Agents without concept A	Agents without concept B
1	72	331	399
2	914	52	399
3	2789	40	399
4–10	2910	40	399
11	3055	29	335
12	4255	20	52
13	6349	20	20

example, take two separate referral networks called RF1 and RF2. The first referral network, RF1 contains consumer agents who are interested in domain 1, the second RF2 contains consumer agents interested in domain 2. Both networks have experts in domains 1 and 2. RF1 should produce similar quality measures as when the concept referral network in this simulation (for the sake of simplicity let us name it CRF) is learning concept A. Additionally RF2 should produce similar quality results as the CRF network in this simulation when it is learning concept B. Learning transitive concepts is no different as is shown in Table 2. The direct and nth best quality metrics are similar when comparing RF1 to the CRF network (and consumer agents are learning concept A) and when comparing RF2 to the CRF network (and consumer agents are learning from concept B).

3.1.3 Accuracy and authoritativeness

Quantitatively similar results are found for this simulation as those found in traditional referral networks with concept knowledge and are left out for the sake of brevity.

3.1.4 Concept propagation

As shown in Table 3, concept propagation occurs quickly (331 agents without concept A for query 1 falls to 40 by query 3) then stalls from query number 4 to 10; agents are at this time deciding if they need to use a concept (if their health has stagnated) and as they adapt their interests to domain 1 they receive the concept relation $1 \rightarrow 2$ from the oracle, which is quickly propagated through the network by query number 13. The 20 remaining agents with neither concept A nor concept B are the 20 experts in domain 2, as they never show interest in a domain for which there is a concept relation.

After receiving both pieces of concept knowledge the consumer agents then set about finding experts in domain 1 and subsequently domain 2. At this point the oracle becomes completely isolated from the network, as it is providing neither answers nor referrals in

domains the consumers are interested in. If the oracle contained a further piece of transitive knowledge, say, the concept relation $2 \rightarrow 3$, this concept would have never propagated to the consumer agents (as it is completely isolated when consumers are showing interest in domain 2). This simulation supports the conjecture that concept knowledge should reside in the experts themselves. In this way no matter how much convergence on experts occurred, the agents would still learn needed concept relations. So, the concept relation $0 \rightarrow 1$ would reside with experts in domain 0, and the concept relation $1 \rightarrow 2$ would reside with experts in domain 1. As agents found experts for domain 0 they would learn about the concept relation $0 \rightarrow 1$, they would then seek out experts for domain 1, where they would learn the concept relation $1 \rightarrow 2$, etc.

It can be concluded that the ability for agents to learn transitively related concepts from a single oracle demonstrates that an oracle can act as a guide to learning. Through successive suggestions an oracle can lead agents to learn about a series of related domains. The need for the oracle to maintain connectivity in the network is also clear.

3.2 Context

This simulation splits the experts into two groups each with expertise in a different domain. Consumer agents are also split into two groups, the first called class A with initial interest in domain 0 but for which searching domain 1 will provide useful answers, the second called class B with initial interest in domain 1 but for which searching in domain 2 will provide useful results. Thus class A agents will find the concept suggestion of $0 \rightarrow 1$ useful while class B agents will find $0 \rightarrow 2$ useful. Two oracles are also present in the network each with concept knowledge relating the same domain (0) to different domains (1 and 2). Thus one oracle is considered a class A oracle, providing relevant concept knowledge to class A consumer agents, and the other is a class B oracle that provides relevant concept knowledge to class B consumer agents.

This simulation has two objectives. First, demonstrate the ability of concept knowledge to distinguish context within or outside of a domain. Second, reinforce the need for a feedback mechanism to determine the utility of concept knowledge.

3.2.1 Health results

Figure 2 plots the average health and the variance in the health metrics of the two classes (A and B) of consumer agents over time. As each class of agent makes use of the correct concept relation and learns in the domain which is useful to them, the health of agents moves to zero (perfect health).

The variance "hump" indicates that this learning is not done all at the same time. Some agents may initially get the correct concept relation, while others do not. Variance in the average health measure for each class of consumer agent will increase as some agents in their class learn and others do not, and then decrease as all agents in the class eventually learn the related domain that is useful to them.

As agents learn which concept is useful to them, they reflect this utility in their trust ratings of the oracle who gave them this knowledge. To demonstrate this fact the trust ratings of a class A oracle are presented in Table 4. The agents in the left hand column represent neighbouring agents that the oracle shared its concept relation with. The numbers 1–5 and 6–12 along the top denote the simulation cycle; the entries in the body of the table denote the trust rating of the agent in the oracle.

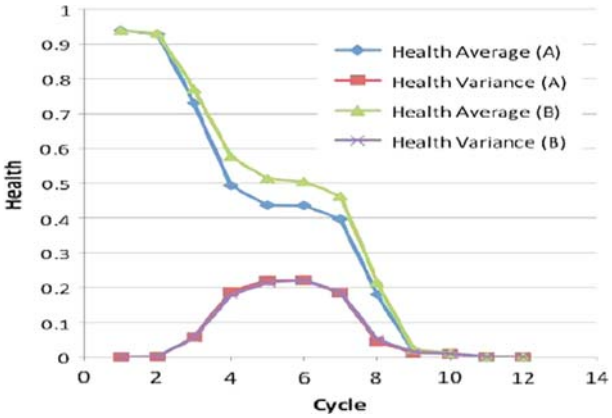

Fig. 2 Health over time

Table 4 Trust ratings of the class A Oracle

Agents	Cycle 1–5	Cycle 6–12
19, 229, 373, 379	0	0.018
95, 146, 283, 353	0	0

As can be inferred from Table 4, agents 19, 229, 373 and 379 are class B consumer agents, and thus after trying the concept relation $0 \rightarrow 1$ from the oracle increased their distrust in the oracle, as it did not provide them with useful results. Agents 95, 146, 283, and 353 are class A consumer agents and find the concept relation $0 \rightarrow 1$ useful and thus do not penalize the oracle. Similar results were obtained for the class B oracle and are not shown here.

Clearly, the ability for agents to locally and subjectively evaluate suggestions allows them to distinguish useful concepts from useless ones and additionally allows them to establish useful oracles within the network. As agents can determine useful context, meaning becomes an emergent property; agents will utilize concept relations which hold the most applicable meaning for their internalized needs.

4 Conclusions

The notion of relational concept knowledge was introduced and shown to effectively allow agents to make use of alternate sources of concept knowledge in a referral network. This is meant to model humans' innate ability to use relational concept knowledge; for instance, when searching online and a certain concept does not provide useful results we then adapt our search based on related concepts. Agents in a referral network are very well adapted to find experts and give referrals; the addition of concept knowledge allows them to leverage this innate ability to learn new and related concepts.

Health was introduced as a mechanism to guide agents in their learning. Just as humans can evaluate the usefulness of a related concept via its application to solve a problem, agents must have a method to evaluate the usefulness of learning. This paper provides a simplified mechanism where an evaluation blueprint is given to the agent; it does not attempt to intro-

duce a mechanism for agents to determine internally if learning a related domain is useful. Furthermore, this paper does not propose how oracles might be constructed. Further work on ontologies or concept maps (Caravalho et al. 2001) may help an agent reason intelligently about useful concepts, just as humans do, but until then agents must be guided by an outside source.

Trust is very important in any sort of social interaction. The local evaluation of concept knowledge based on an internal feedback mechanism allows agents to determine the utility of shared concepts. This utility used as trust allows agents to determine useful providers of concept knowledge based on their needs. It acts to prevent malicious or useless concept knowledge providers from continually misguiding consumer agents in the network.

Propagation of concept knowledge ensures that changing meaning can quickly move through the network in a way that is similar to the way in which new subjects, fads or crazes "move" through human society.

Acknowledgments We would like to acknowledge Pinar Yolum and Munindar Singh for providing the simulation test bed on which this research was developed.

References

Caravalho M, Hewett R, Canas AJ (2001) Enhancing web searches from concept map-based knowledge models. Institute for Human & Machine Cognition, Pensacola, FL

Page L, Brin S, Motwani R, Winograd T (1998) The PageRank citation ranking: bringing order to the web. Stanford Digital Library Technologies Project, Stanford

Salton G, McGill MJ (1983) An introduction to modern information retrieval. McGraw-Hill, New York

Yolum P, Singh MP (2002) An agent-based approach for trustworthy service location. Lecture Notes in Artificial Intelligence, pp 45–56

Yolum P, Singh MP (2003) Emergent properties of referral systems. In: Proceedings of the Second international joint conference on autonomous agents and multiagent systems, pp 592–599

Yolum P, Singh MP (2004) Engineering self-organizing referral networks for trustworthy service selection. IEEE Trans Syst Man Cybern 35:396–407

Yu B (2001) Emergence and evolution of agent-based referral networks. Dissertation, Department of Computer Science, North Caroline State University