AN ONTOLOGY AND CONCEPTUAL GRAPH BASED BEST MATCHING ALGORITHM FOR CONTEXT-AWARE APPLICATIONS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

REZA KOUSHAEIAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF INFORMATION SYSTEMS

MAY 2011

Approval of the Graduate School of Informatics

____

Prof. Dr. Nazife Baykal
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

____

Prof. Dr. Yasemin Yardimci Cetin
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

____

Assist. Prof. Dr. Altan Koçyiğit
Supervisor

Examining Committee Members

Prof. Dr. Nazife Baykal	(METU, II)	______________________
Assist Prof. Dr. Altan Koçyiğit	(METU, II)	______________________
Dr. Ali Arifoğlu	(METU, II)	______________________
Assist Prof. Dr. Erhan Eren	(METU, II)	______________________
Assist Prof. Dr. Cüneyt Sevgi	(IŞIK University)	__________________

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Reza Koushaeian

ABSTRACT

AN ONTOLOGY AND CONCEPTUAL GRAPH BASED BEST MATCHING ALGORITHM FOR CONTEXT-AWARE APPLICATIONS

Reza Koushaeian
M.Sc. Department of Information Systems
Supervisor: Assist. Prof. Dr. Altan Koçyiğit

May 2011, 95 Pages

Context-aware computing is based on using knowledge about the current context. Interpretation of current context to an understandable knowledge is carried out by reasoning over context and in some cases by matching the current context with the desired context. In this thesis we concentrated on context matching issue in context-aware computing domain. Context matching can be done in various ways like it is done in other matching processes. Our matching approach is best matching in order to generate granular similarity results and not to be limited to Boolean values. We decided to use Ontology as the encoded domain knowledge for our matching method. Context matching method is related to the method that we represent context. We selected conceptual graphs to represent the context. We proposed a generic algorithm for context matching based on the ontological information that benefit from the conceptual graph theory and its advantages.

Keywords: Context, Context Aware Computing, Ontology, Conceptual Graph, Context Matching

ÖZ

BAĞLAM FARKINDALIKLI UYGULAMALAR İÇİN ONTOLOJİ VE KAVRAM ÇİZGESİ TABANLI EN İYİ EŞLEŞTİRME ALGORİTMASI

Koushaeian, Reza
Yüksek Lisans, Bilişim Sistemleri Bölümü
Tez Yöneticisi: Yar. Doc. Dr. Altan Koçyiğit

Mayis 2011, 95 Sayfa

Bağlam farkındalıklı hesaplama mevcut bağlam hakkında bilginin kullanımına dayanmaktadır. Mevcut bağlamın anlaşılabilir bilgi haline getirilmesi bağlam hakkında akıl yürütmeyi ve bazen de istenen bağlam ile mevcut bağlamın eşleştirilmesi ile yapılır. Bu tezde, bağlam farkındalıklı hesaplama alanındaki bağlam eşleştirme problemi üzerine yoğunlaştık. Bağlam eşleştirme diğer eşleştirme süreleçrinde olduğu gibi çeşitli yollarla yapılabilir. Bizim eşleştirme yaklaşımımız boole sonuç üretme yanında ince kırılımlı benzerlik sonuçları oluşturmak için en iyi eşleştirmedir. Eşleştirme yöntemimizde kodlanmış alan bilgisi olarak ontoloji kullanmaya karar verildi. Bağlamı temsil etmek için ise kavramsal çizgeler seçildi. Bağlamsal çizge teorisi ve onun avantajlarından yararlanan ontolojik bilgiye dayalı bir genel algoritma önerildi.

Anahtar Kelimeler: Bağlam, Bağlam Farkındalıklı Hesaplama, Ontoloji, Kavramsal Çizge, Bağlam Eşleştirme.

ACKNOWLEDGEMENT

First of all , I would like to state my sincere gratitude to my advisor Assistant Prof. Dr. Altan Koçyiğit. His guidance helped me so much during the research and writing of this thesis. He made a great contribution into my thesis and supported me during the research.

I would like to express my deepest gratitude to my wife for their love, encouragement, and support throughout my education. This thesis would never have been completed without their love and endless support.

This thesis is dedicated to Pardis.

TABLE OF CONTENTS

LIST OF TABLES

TABLE OF FIGURES

LIST OF ACRONYMS

RDF	Resource Description Language
CG	Conceptual Graph
CoBrA	Context Broker Architecture
SOCAM	Service-Oriented Context-aware Middleware
CASD	Context-aware Service Discovery
SOA	Service Oriented Architecture
CGIF	Conceptual Graph Interchange Form
KIF	Knowledge Interchange Format
DF	Display Form
LF	Linear Form
GPS	Global Positioning System
ICCS	International Conferences on Conceptual Structure
OWL	Web Ontology Language

CHAPTER 1

INTRODUCTION

Context matching is one challenging area in context-aware applications domain. In this thesis, we propose a generic approach for context matching that can be used in different application domains. In order to fulfill this task we use the Conceptual Graph to represent the context. We benefit from the Ontology advantages to represent domain knowledge in a formal way. We employ conceptual graphs to represent the Ontology formally. Our proposed Context matching algorithm is based on conceptual graph matching.

1.1 Context-aware computing

Context-aware computing is one aspect of pervasive computing and it was coined by Weiser in 1991. Context-aware systems have the ability to be aware of their environmental context. The advantage of this feature is the ability of context-aware systems in adapting their behavior with the current context automatically. This augments usability and effectiveness of application (Baldauf, Dustdar, & Rosenberg, 2007). There are some overlaps between definitions of context-aware and sentient computing. Context-aware computing is similar to sentient computing but it is agent based. In context-aware computing the agents build their environmental model whereas sentient computing uses sensors to be conscious of its environment (Poslad, 2009)

One challenging side of the context awareness is continually changing environment (B. Schilit, Adams, & Want, 2008). In developing context-aware applications respecting to the distribution, modifiability and reusability aspects is another challenge for developers. In addition, limited options in rapid design and development of context-aware applications are related to lack of well-defined conceptual models and frameworks. (Corbett, 2004)

1.2 Notion of Context

There are two viewpoints on context studies. The first one is cognitive science view that consider context in order to model situations and focus on human behavior as the main key for generating this model. The second perspective is engineering view that is concentrated on problems that should be solved with the help of context in order to reason or represent knowledge in problem domain (Brézillon, 1999).

The engineering view is dedicated to external context or physical context which includes context data of physical environment collected by physical sensors (J.-yi Hong, Suh, & Kim, 2009). Cognitive context has an important role in developing personalized context-aware applications based on user's preferences, task and emotional state of user, the cognitive domains, such as information retrieval, decision making, situation monitoring, and so on. (J.-yi Hong et al., 2009)

Brezillon claimed that these two viewpoints are slightly different, and the opposition between them is originated from how they consider context, while cognitive view emphasizes the level of interaction between agents, whereas the engineering view considers the knowledge representation level.(Brézillon, 1999)

The comprehensive definition of context that we will use in this thesis is the definition of Dey: "Context refers to any information that characterizes a situation related to the interaction between users, applications, and the surrounding environment." (Dey & Abowd, 2000)

1.3 Context representation methods

Major difficulties in processing context information are originated from ill definition of context and the way used to represent it. Selecting an appropriate method is important because our final approach in context matching depends on context representation method used. A proper representation method should satisfy following criteria:

- Support context variations
- Suitable for available context acquisition mechanism.
- Ability to describe ambiguous contexts
- Easy to use and understand

We do not consider challenges in capturing context information and checking its accuracy in this study. We assume that incoming context information is accurate and in ideal format. It is important to separate capturing context information from how we use them. This separation of concerns contributes to the generality of our algorithm. The importance of this separation is the main idea of developing conceptual frameworks that guarantee separation of concerns in capturing the context from the publishing and using the same context by the application.

1.4 Context matching

Context matching is a matching process used in context-aware applications. Matching has various usages in different research fields. Various matching methods have been developed to satisfy specific needs, such as pattern matching, exact matching, best matching, fuzzy matching, string matching and record matching. The matching process deals with two sets of information in order to compute the similarity between them. The similarity calculation is highly interconnected to distance calculation that means figuring out how dissimilar are two parts. (KOCABALLI, 2005)

In context-aware applications the context matching process is helpful in order to set the context-aware applications' behavior according to specific conditions. The objective of matching process in this work is matching the current sensed context with the set of predefined contextual information, namely desired context. The obtained result indicates similarity of current context with desired context.

1.5 Conceptual graphs

Conceptual graph model was developed by Sowa (J. F. Sowa, 2000), and further works in this area are done by the conceptual graph community. The roots of conceptual graph can be found in natural language processing, semantic networks, database and logics, and especially the existential graphs of pierce. The conceptual graph (CG) is a family of formal definitions that have roots in Sowa's initial works. It consists of basic conceptual graphs, simple conceptual graphs, positive nested conceptual graphs and full conceptual graph (Chein & Mugnier, 2008). A brief overview of conceptual graphs is given in Chapter 2.

1.6 Ontology

Ontologies are used in various research areas like problem-solving methods, domain-independent applications and software agents. In most of the cases, developing Ontology is not the ultimate goal instead it is used as a tool in applications (Noy, McGuinness, & others, 2001). Adding ontologies with automated reasoning bring intelligence to applications in order to act more similar to human beings. Applications can benefit from this aspect of Ontology in the areas of decision support, natural language understanding, knowledge management, and intelligent databases.

1.7 Context Matching based on the Conceptual Graphs and Ontology

Selecting an appropriate matching technique is highly dependent on the context representation method. We represent the contextual information including both current context and desired context as conceptual graphs therefore our matching approach is based on conceptual graph matching.

Conceptual graphs are an appropriate tool for visualizing the contextual information. Using graphs as a graphical representation tool brings some advantages in making, editing and revising knowledge and also permit a natural representation (Corbett, 2004). In addition, available experiments in the field of conceptual graphs should be mentioned as another advantage.

The knowledge of domain including the concepts and relations are also represented as conceptual graphs. The CG vocabulary consists of the concept hierarchy and relation hierarchy and the populated ontology that describes the concepts and their interrelations represented as the CG. We keep the long-term context information in the ontology. In contrast the short-term sensory data is saved as the context observation units. Context observation units are also CG with four main relations in order to support contextual information such as hour, date, location and status of one actor.

In this thesis, we proposed an easy to use and easy to implement Ontology based context matching method that is based on the usage of conceptual graphs proposed in the literature. The algorithm devised matches current context with a set of desired contexts and produces a similarity score for each desired context. We tried to build a generic matching algorithm which can be used in different context-aware applications. We represented the knowledge base of the domain as an Ontology and benefit from the advantages of using ontological representations.

1.8 Thesis organization

In Chapter 1, the subject of this thesis study is summarized and the problem, the objectives and the scope of the thesis is presented.

In chapter 2, we firstly review different definitions of context and then context dimensions are introduced. Context-aware computing applications are summarized. Context matching is defined and Ontology-based context matching is introduced. Then, conceptual graphs are introduced and some related works on conceptual graph matching are presented.

In chapter 3, we introduce our conceptual graph based context matching method. At the beginning we explain how conceptual graphs can be used to represent context and Ontology, and then a method is proposed to augment query CG to obtain the desired context CG. In addition we propose a method to filtering the context observation units and generating appropriate current context CG based on the Ontology and desired context CG. Finally we compute the similarity between desired context CG and current context CG based on their relation, concept type and referent similarity based on the Ontology and we will present the best matching algorithm.

In chapter 4, we investigated the applicability of proposed algorithm through the use of a cinema scenario. The results obtained by applying the proposed algorithm is also compared with the results obtained from previous similar works.

In chapter 5, a brief summary of the study is given and salient features of the proposed matching method are discussed. Finally directions for future works in the context matching based on ontological information are suggested.

CHAPTER 2

BACKGROUND AND RELATED WORK

2.1 Context definition

Context, in thesauruses, has similar definitions as ambient, background, whole picture and total situation. Definitions of context contained many aspects of context aware computing in research community.

Schilit et. al. mentioned the location information as the necessary context to adapt a context-aware application to its location of use, in addition located-objects like printers, terminals and nearby people and their related changes (B. N. Schilit & Theimer, 1994).

Brown et al. considered context as user's context, mainly focused on user's location information and in general any tailored user's information like location, time of day, season of the year, temperature and so forth. (P. J. Brown, Bovey, & Xian Chen, 1997)

Ryan et al. defined context as information about computational environment "such as location, time, temperature or users' identity" (Ryan, Pascoe, & Morse, 1998).

Schmidt et al. defined context as the description of current situation on an abstract level, his definition mainly focuses on current situation in order to add smartness to applications, and he defined architecture for context recognition, regarding to three dimensions named: Self dimension that includes device state, physiological, cognitive, Environment dimension that supports both physical and social, Activity dimension that works on behavior and task (Schmidt, A. and Aidoo, K. and Takaluoma, A. and Tuomela, U. and Van Laerhoven, K. and Van de Velde, W., 1999).

Chen et al. defined context as the set of environmental states and settings that either determines an application's behavior or in which an application event occurs and is interesting to the user. (Chen, G. and Kotz, D., 2000)

Dey et al. extended the context definition such that it is not limited to user's context. Information the user that is attending to, emotional state, focus of attention, location and orientation, date and time of day, objects and people in the user's environment constitutes the context (Dey, 1998). Dey et al. proposed a complete definition of context as any information that characterizes a situation related to the interaction between users, applications and surrounding environment.(Dey, Abowd, & Salber, 2001)

2.2 Context dimensions

Dey's definition of context support various dimensions including time, people, location, status, things etc. whereas previous works consider specific dimensions that are related to the domain of interest. He proposed a context categorization in order to handle context diversity in a systematic way. He emphasized three contexts categories as the major context groups that are used in context-aware applications: "place: geographical spaces like rooms, offices, buildings or streets, people: individuals or groups and things: physical objects or software components". In addition, he listed common characteristics of different contexts as time: help to augment historical information, location: help to add detailed information about geographical coordination's or relative spatial relations, ID: a unique identity in domain and status: describes sensible mood of the entity (Dey et al., 2001).

Tao Gu et al. (Gu, Pung, & Zhang, 2005a) considered location, activity, person and computational device as general contexts and brought them in generalized context Ontology. Kocaballi(KOCABALLI, 2005) in the same way, defined four characteristics for context information namely location, people nearby, time and status. Harry Chen et al. (Chen, Finin, & Joshi, 2003) considered place, agent, agent's location and agent's activity as contexts in their proposed Ontology for pervasive context-aware systems, named COBRA-ONT.

What Dey considered as context, supports a wide range of contexts and is extendable in order to add advanced meanings to context. People context category supports human agents in CoBrA (Context Broker Architecture) (Chen et al., 2003), things category supports computational tools in SOCAM (Service-Oriented Context-aware Middleware) (Gu, Pung, & Zhang, 2005a) and software agents in CoBrA. In addition, context dimensions that were mentioned in his work enrich the meaning of context. By defining characteristics for each context in the first step we add more detail to a context that provides a better understanding of the context e.g. location of a place context can be defined as the place's GPS position. Beyond the explanation side of context characteristic, some aspects of fuzziness can be extracted from contextual information. For example, suppose that a combination of place context like [place: shopping center] and hour context such as [hour:3:00 am] is given .The meaning of this situation is not the same as the combination of a place context of shopping center with a hour context that describes a regular work hour like 10:30 am, so by adding the hour to place context we add new meaning to place context. In this example, the objective of being in or near a shopping center at early morning hours might be interpreted as spending time at a night club or withdrawing money from an ATM station nearby. In the same way, a specific time can take different values in different days and different places. Breakfast status context with the place context of hotel in a date dimension of holiday has a different hour from the Breakfast status with home place dimension and regular working day context.

We found Dey's definition for context categories and context dimensions comprehensive and we consider location, time of the day (hour), date and status as the general context dimensions in this study.

2.3 Categories of context–aware computing

Schilit(B. Schilit et al., 2008) remarked major concerns of the context awareness as where the user is, who the user is with, what resources are nearby. He also proposed four categories of context-aware applications by the product of nature of context-aware application which are getting information or doing a command along the application's treatment that is manual or automatic, as shown in Table 1 These four categories are studied in specific case of PARCTAB (B. Schilit et al., 2008), but in general they can be assumed as a good categorization of context-aware applications.

Table 1 Four categories of context-aware applications based on Schilit's work (B. Schilit et al., 2008)

	manual	Automatic
Information	Proximate selection & contextual information	Automatic contextual reconfiguration
command	Contextual commands	Context-triggered actions

A brief interpretation of each category in context-aware applications can be:

- Proximate selection & contextual information: proximate selection is a user interface technique that nearest objects are announced to the user. The represented information can be sorted based on the proximity or alphabetically.
- Contextual information in similar way depends on the current location of the user. The main idea is that people's behavior in certain locations is predictable. The system generates results of contextual information queries based on the context of their activation time. (B. Schilit et al., 2008)
- Automatic contextual reconfiguration: reconfiguration is the process of configuring of new settings like adding a new connection or removing an old one. The main challenge is how to use contextual information in order to reconfigure a set of systems or environment. For example, entering a room should make automatic binding between the user's digital pad and room's available devices like printer. (B. Schilit et al., 2008)
- Contextual commands: similar to contextual information category the contextual command uses the same fact that users do certain things in certain contexts. For example, we do special things in certain places like library, cafeteria or classroom. Contextual commands target this fact by automatically setting the parameters of a command. For example, the print command might, by default, print the nearest printer. (B. Schilit et al., 2008)
- Context-triggered actions: In this category current context triggers a specific command. The structure of such commands is similar to IF-THEN conditions that tailor context-aware system's behavior. (B. Schilit et al., 2008)

Dey et al. (Dey, 2000) proposed the context-aware application categories based on how applications use the context. He categorized context-aware applications as:

1) Presenting information and services: these applications that present context information to the user or recommend a service to the user based on the context information. Some examples in this category are:

- Showing user's location on a map and suggesting interesting places.(Abowd et al., 1997),(Bederson, 1995),(Davies, Mitchell, Cheverst, & Blair, 1998),(Feiner, MacIntyre, H\öllerer, & Webster, 1997),(Fels et al., 1998),(McCarthy & Anagnost, 2000),(McCarthy & Meidel, 1999)
- Suggesting the closest resource to the user (B. Schilit et al., 2008)
- Recognizing and showing the in/out information for a group of users(Dey, Abowd, & Salber, 1999)
- Presenting ambient information (Heiner, Hudson, & Tanaka, 1999),(Ishii & Ullmer, 1997),(Mynatt, Back, Want, Baer, & Ellis, 1998),(Weiser & J. S. Brown, 1996)
- Providing information about receiver's situation before setting up a call (Schmidt, Takaluoma, & M\äntyj\ärvi, 2000)

2) Automatically executing a service: these applications invoke an action or update configurations by changing context in automatic manner. Some examples are:

- A telephone call routing system that follows user when he move from one workstation to another workstation(Want, Hopper, Falcao, & Gibbons, 1992)
- Car navigation system that reconfigures driving path when driver lose a turn(Hertz, n.d.)
- Whiteboard capture application recognizes that opportunistic meeting happens and starts to capture it.(Brotherton, Abowd, & Truong, 1998)
- Augmented handheld devices with sensors that determines context of use and change the setting and actions (Harrison, Fishkin, Gujar, Mochon, & Want, 1998),(Hinckley, Pierce, Sinclair, & Horvitz, 2000),(Schmidt, A. and Aidoo, K. and Takaluoma, A. and Tuomela, U. and Van Laerhoven, K. and Van de Velde, W., 1999)
- A wearable camera that is also computer and sensing device that capture and save images in both conscious and unconscious situations based on predefined factors (Healey & Picard, 2002)
- Devices that link personal information to locations and run reminders in specific locations. (Beigl, 2000),(Marmasse & Schmandt, 2000)

3) Attaching context information for later retrieval: these applications mark captured data with the contextual information. Some examples are:

- An application for observation that can add location and time to the saved information (Pascoe, Ryan, & Morse, 1998)
- Access to meeting notes by knowing people in meeting, time of meeting and location of meeting(Brotherton et al., 1998), similar to this(Rekimoto, 1999) and (Dourish et al., 2000)
- Forget-Me-Not (Lamming & Flynn, 1994) and Remembrance agent (Rhodes, 1997) as applications that attach context to files in order to easier retrieval.

Chen et al. (Chen, G. and Kotz, D., 2000) proposed two categories to context-aware computing:

- Active context awareness: An application automatically adapts to discovered context, by changing the application's behavior. The main focus of this category is running a command.
- Passive context awareness: An application presents the new or updated context to an interested user or makes the context persistent for the user to retrieve later. This category of application aim to inform or memorize.

2.4 Context matching

The matching process considers two sets of context information. The first one is the current context that describes current situation and consists of several contextual information pieces, all related to each other. The second one is called desired context that describes the desired situation.

Interpreting the results of matching depends on our ultimate need. If we are interested in exact matching it is necessary to define a similarity threshold in order to say the current context matches to the desired context. In best match approach the provided results are ranked and the highest similarity score shows the best matching option.

The output of matching process in best match approach is a similarity score. This score tells us how much the current context is similar to the desired context. In general, the score takes a value between 0 and 1. Where 0 indicates no similarity and 1 indicates the maximum similarity.

2.5 Some Context Matching Related Works

Kocaballi (KOCABALLI, 2005) proposes a weighted granular best matching algorithm and focuses on four major context elements namely people, location, time and activity. The objective of matching process in this work is matching the current sensed context with the predefined contextual situation in order to trigger some actions. Location matching is based on a six level granular tree. Depending on the distance between two contexts the level of similarity is defined as fixed values for each level. For example a target context located in place A is coded as 113161 and the desired location of B coded as 112 have two common digits and located in level 2 of similarity. The same approach has been chosen for activity matching but in two level of granularity. People contexts match in an exact matching manner. Time context considers a composite format that includes both time and day. Total time similarity factor used to match the time in a fuzzy matching approach. Finally each dimension has a weight. The weighted score of each dimension calculated regarding to dimension similarity score and its weight. Total matching score is equal to the total of dimensions' (location, people, activity, time) weighted score.

In (Pawar & Tokmakoff, 2006) CASD (Context-aware Service Discovery) is proposed. CASD try to understand the appropriate service for the client. CASD system performs a basic matching based on keywords in query information sent by the clients. After selecting basic services the CASD system requests for current context of basic services. The collected contexts from basic services are stored in service graph. After that CASD system ask for the client context. In the cases that a persistence service recovery requested, the CASD system subscribe to the context change event of the client. Client's contextual information and service graph are sent to the reasoner. The reasoner matches the client information with services in service graph and returns a service. Finally, CASD system sends the service reference to the client.

CASD has ability to support continuous matching that is a main advantage for nomadic users. In basic matching the CASD system subscribe to the selected basic service in order to be informed about the contextual changes, also the result of the basic matching is kept in a database for further investigation and also reasoner looking for any better matching results to inform the CASD system.

2.6 Conceptual graphs

John Sowa (J. F. Sowa, 2000) introduced Conceptual graphs (CGs) based on the Sanders Peirce's existential graph notion (Van Harmelen, Lifschitz, & Porter, 2008), which are suitable representation of meaning. CGs have some roots in semantic networks.

Conceptual Graph has some advantages in knowledge representation such as:

- It is a human readable and also computer readable representation
- It can express logical meaning in a precise format
- It has a close representation to natural languages
- There is an international community, ICCS working on this topic.

Conceptual graphs are represented in three forms:

1. DF or Display Form that is a graphical representation
2. CGIF or Conceptual Graph Interchange Form that is a logically equivalent representation in predicate calculus and in Knowledge Interchange Format (KIF) which is a notion for representing logic. The CGIF has restricted character set with a simpler syntax than KIF.
3. LF or Linear Form that is a compact and readable format

The DF and the LF representation forms are designed and used for communication between human and computers. The CGIF format is used for communication between computers. In IT systems CGs are used for internal representation. For communication between systems that use other internal representations, KIF can be used. CGIF can be translated into a KIF for further logical inference (J. F. Sowa, 1999). Knowledge Interchange Format (KIF) is "a language designed for use in the interchange of knowledge among disparate computer systems (created by different programmers, at different times, in different languages, and so forth)" (Genesereth, 1998).

Conceptual graphs are used as a representation method for logic based on semantic networks. A CG has two kinds of nodes, namely concepts and conceptual relations. Also, there are some arcs that connect concepts and relations. Nodes of CG are illustrated as:

1) Concepts are represented by boxes in DF or square brackets in LF.

2) Conceptual relations are represented by circles in DF or parentheses in LF.

The conceptual graph is a bipartite graph (or bi-graph). The bipartite graphs' edges connect two vertices that are in two disjoint sets. So we will not see an arc from a concept to another concept. As an example, we translate the "A cat is on a mat" to the DF of a CG as shown in Figure 1.

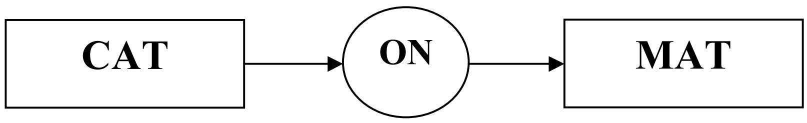

Figure 1 A simple CG

The same expression can be written in linear form as [CAT]®(ON)®[MAT]. The generally written [Concept]®(Relation)®[Concept] should be read like: [Concept]has a(Relation)which is [Concept]. In reverse mode we can write the same statement in the form of: [Concept]¬(Relation)¬[Concept] that means: [Concept]is a(Relation)of[Concept],As another example, [Walk]®(Agent)®[person :John] is read as walk has an agent which is person :John. [person :John]¬(Agent)¬[Walk] is read as person :John is an agent of walk.(Humanistic Informatics at Department of Communication, Aalborg University, Denmark, n.d.)

A concept consist of two parts a concept type and a referent. The concept [person :John] has a concept type Person and of referent John.

Relations also have a relation type. The relation type is denoted by T_R. The number of arcs that belongs to a relation shows the valence (or arity) of a relation. Based on the relation arity T_R can be partitioned into k sets $(T^i{}_R , \leq)_{i=1..k}$ of relation types of arity i where $1\leq i \leq k$, k shows the relation arity. Signature function can be used to find out the arity of a relation and the sequence of concept types in a conceptual relation. The B symbol is used to show relation signature.(Chein & Mugnier, 2008)

$$B{:}T_R \rightarrow B(t_r)\ ,\ B\ (t_r) \in \underrightarrow{T_C \times \ldots \times T_C}\ _{\text{k times}} \quad B\ (t_r)=\{t_{c1}, \ldots , t_{ck}\}$$

Here we have some examples to clarify the meaning of relation arity:

Consider the CG in Figure 2 the Betw relation has a signature of three concept types

B (Betw)={Room, Room, Room}

The similar relations with signature of three concepts called Triadic. A relation with one arc (signature of one concept) is called Monadic and relations with two arcs (signature of two concepts) are called Dyadic.

All the concepts that are related to the relation should have a sequence, we use number tags over relation arcs and last placed concept in relation shows the last sequence.

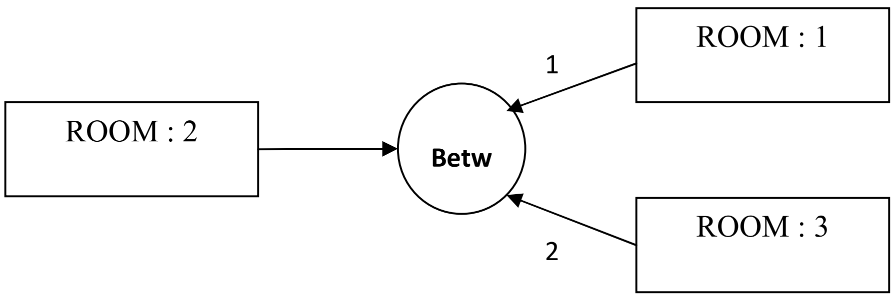

Figure 2 A CG relation

As described above CGs contain of two types of nodes: concepts and relations. Two sets keep the related items for these node types. The first set is T_C includes concepts types and the second one is T_R that contains relation types. Both of these sets are ordered by a sub-typing relation that also called specialization relation or a-kind-of relation. This relation between type items is denoted by $\leq$ symbol. If we have "t_2 as the specialization of t_1" we write $t_2 \leq t_1$.This statement can be read as "t_1 is a generalization of t_2", this also means that any entity of type t_2 is also of type t_1(Chein & Mugnier, 2008). In the type hierarchy the most general type is represented by the $\top$ symbol.

In CG vocabulary there is another set, I which is the set of individual markers. Concepts are representation of entities in a specific domain. Concepts with referents are specific entities namely individual concepts. Unidentified concepts are called generic concepts. We use * symbol as the generic marker in order to represent unspecified entity regardless of its type. Individual concepts can be referenced by individual markers. The set of markers is defined as M= I ∪ {*}

.We use this notation to represent a generic concept t, (t,*),[t] and to represent the individual concepts (t, m), [t:m]

2.6.1 Definition 1: Vocabulary

Vocabulary is a triple (T_C, T_R, I) where: (Chein & Mugnier, 2008)

- T_C and T_R are finite, disjoint sets.
- T_C is the set of concept types and partially ordered by a relation $\leq$ and has greatest value represented with $\top$.
- T_R is the set of relation symbols, that is ordered by a relation $\leq$ and is partitioned into $T^1{}_R$, … ,$T^k{}_R$ sub-sets of arity 1… k
- I is the set of individual markers which is disjoint from T_C and T_R. Two distinct individual markers represent distinct entities. Set of markers is defined M= I ∪ {*}. The generic marker (*) is greater than any element in I, and non-pair elements in I are incomparable.

Two functions are usually considered as the extension of a vocabulary (Chein & Mugnier, 2008):

1) Individual typing function (τ): is used to map the individual markers to the concept type. That is, individuals can be typed by this function. The function returns the category of a referent, that is $\tau: I \rightarrow T_C$ .Typing function guarantees CGs to be well-formed. In other words if we have a relation: $r\ (t_{C1}:m_1 , \dots, t_{Cn}:m_n)$ then $\tau(m_i) \leq t_{Ci} \quad 1 \leq i \leq n$

As an example if there is a relation Poses<Person, Object> and we have Poses <friend: Reza, Object: gift> this relation is validated when $\tau\ (m_i) \leq t_{Ci} \quad 1 \leq i \leq n$. If we assume τ(Reza)= friend then this relation is validated if in relation hierarchy we should have friend≤ Person.

2) Relation symbol signature: By using the signature we specify all concept types that can be used in a relation. We can define two relations as signature compatible if:

$$\forall\ r_1,r_2 \in\ T^i{}_R \quad r_1 \leq r_2 \Rightarrow B(r_1) \leq B(r_1)$$

2.6.2 Definition 2- labeling functions:

Labeling function is used in order to assign a label to the nodes and edges in a graph, which is defined as: (Chein & Mugnier, 2008)

- A Concept node c is labeled by a pair (type(c), referent(c)) where type(c)$\in T_C$ and referent(c)$\in$ I U {*}
- A Relation node r is labeled by l(r)$\in T_R$, l(r) means type of r and can be represented as type(r).
- Edges that belong to a relation r have an order that starts from 1,… to B(type(r))

l(c) is the label of the node c of a CG from the vocabulary:

$\forall$ r$\in$ R , l (r)$\in T_R^{B(r)}$ $\forall$ c$\in$C , l (c)$\in T_C \times$(I U {*}) , B(r)=<$c_1, c_2 \ldots, c_n$> such that

l (r)=type(r) and l (c)=(type(c), reference(c)) then type(c_i) $\leq$ B(type(r))$_i$

_ B (type(r))$_i$ means the i-th argument of relation r<$t_{C1}, \ldots t_{Cn}$>

2.6.3 Definition 3- Canon:

Canon is a tuple (T , I , $\leq$, :: , B) that defines the domain of the conceptual graph where (Corbett, 2003)

- T is the set of types that includes two disjunctive sets of T_C for concept types and T_R for relation types.
- I is set of individuals that can exist.
- $\leq$ is the subtype relation
- :: is the conformity relation that relate type labels to individual marker to ensure that related type have a meaning in domain.
- B is the canonical basis function that associates each relation type with the concept types that may be used with the relation.

2.6.4 Definition 4 – Conceptual Graph:

Conceptual graph is a 2 tuple CG=(Canon, G), where:

- Canon=(T , I , ≤ , :: , B)
- G=(C, R, E, l) that is a ordered bipartite graph that:
  - C is set of concepts.
  - R is set of relations.
  - E is set of edges.
  - l is labeling function.

2.7 Ontology

Ontology is an explicit, formal representation way that describes the already existing or potentially existing concepts and interrelationships in a machine readable format (Noy et al., 2001).

Ontology encodes knowledge in a domain. Domains are restricted information areas that are concentrated on a specific subject like education, finance, medical care, etc. The concepts and the relationships among them are described in Ontology in a computer usable method. Ontology can be represented in various structures from the simple taxonomy, to metadata schema, to logical theories (Heflin, 2004). Regarding context matching, that is the main interest in this work, the Ontology representation method should be able to represent context (in its general meaning), the relationship between contexts and the properties of the context in a consistent way. Ontology is an approach to structure and define the pertinent metadata. The Ontology represents semantics of contexts and enables applications to use these semantics.

Major advantages of using Ontology are:

- The structure of knowledge is generated and can easily be shared by the users
- The generated Ontology is reusable
- Assumptions related to the domain become explicit
- The domain knowledge is isolated from the operational domain
- The generated knowledge is normalized
- It reduces the hard coding attempts to represent knowledge

2.7.1 Principles of designing Ontology:

There are important principles that should be considered during the design of Ontology. In (Korpipaa & Mantyjarvi, 2003), goals and design principles for the Ontology for the mobile device sensor-based application is summarized as following:

1. Domain
2. Simplicity
3. Practical access
4. Flexibility
5. Facilitate inference
6. Genericity
7. Efficiency
8. Expressiveness

2.7.2 Formal definition of Ontology based on the conceptual graph theory:

Conceptual graphs are suitable for representing Ontology. CGs are also appropriate for comparing, merging and performing some other operations on Ontology. CG theory already provides tools, structure and formal definitions for these functional features (Corbett, 2004). In addition, there is no difficulty in using CGT for representing Ontology where there are reasoning, subsumption, inheritance and comparison functionalities defined for types and individuals in a conceptual graph. These available mechanisms can be helpful in reasoning process using Ontology when it is represented with a CG by considering types as objects. In other words, the CG formed Ontology will automatically present these functionalities that are originated from CGT. It also provides a definition of semantic in forms of hierarchy and subsumption (Corbett, 2004).

Representing the Ontology with conceptual graph is an adaptable contextual information representation method that can be applied to different scenarios. Therefore, it may be helpful to propose a generic algorithm. When the Ontology is represented with a CG, based on its formal definition it provides a concept type hierarchy that can be used as a framework to define other conceptual graphs that conform to the hierarchy of concepts (Corbett, 2004).

In this thesis, we use formal definition of Ontology that is based on conceptual graph theory. Conceptual graphs and canon are defined in previous sections. The canon can be considered as the set of CGs that is well-defined and significant in the domain of knowledge (Corbett, 2008).

2.7.2.1 Definition 5-Formal definition of Ontology (Corbett, 2003)

An Ontology is a 5-tuple O=(T_{CM},T_{RM}, I_M ,≤, :: , B) that is constructed based on Canon definition as:

- T_{CM} is the set of concept types for domain M.
- T_{RM} is the set of Relation types for domain M
- I_M is the set of individuals in for domain M
- ≤ subtype relation
- B is canonical basis for each relations

In (P. Nguyen, Kaneiwa, Corbett, & M. Q. Nguyen, 2008) the original definition of Ontology is extended to reach a complete Ontology. In this "Closure of the original Ontology" the concept types and relation types are extended in order to supply some missing arguments or properties of relations. Extended relation is produced by merging arguments of its sub-type relations to build a super-type relation.

2.7.2.2 Definition 6- relation extension (P. Nguyen et al., 2008):

Assume relation r_i with arity(r_i)=n for which B(r_i)=($t_{Ci\ 1}$,….,$t_{Ci\ n}$) and relation r_j arity(r_j)=m for which B(r_j)=($t_{Cj\ 1}$,….,$t_{Cj\ m}$) , r_i is said to extension of r_j and written as r_i=ext(r_j) if all arguments of r_j are presented in r_i ;

r_i=ext(r_j) if ($t_{Ci\ 1}$,….,$t_{Ci\ n}$)= ext ($t_{Cj\ 1}$,….,$t_{Cj\ m}$) which means $m \leq n$, $\forall\ 1 \leq k \leq n$ $t_{Ci\ k}= t_{Cj\ k}$

The new Ontology with relation type hierarchy has broadened components as:

- I is the set of individuals (or instances) include all concept types and relation types $I=I_C \cup I_R$
- conformity function is extended to support relating relation instances to relation types conformity (::) : $I_C \cup I_R \rightarrow T_C \cup T_R$
- Function B is extended in order to define arguments for both T_R and I_R sets.

$$B: T_R \cup I_R \rightarrow \tau(T_C) \cup \tau(I_C)$$

The described Ontology based on CG formalism, consists of a type and relation hierarchy that works as the framework of Ontology. The type hierarchy also can be used as a simple Ontology that is mentioned in (Corbett & Mayer, 2005).

Another aspect of Ontology in addition to defining a framework is acting as knowledge container. In this thesis, long term concepts and relations are kept in the Ontology as the knowledge base used in the matching process. In CG formalism, the framework of knowledge supports type and relation hierarchy and the knowledge of domain is represented in the form of a specific conceptual graph. In this specific CG we represent the long term knowledge about the facts or situations. The specific concepts or individuals should be considered as new concept types and added to the type hierarchy. This guaranties the graph derivation through projection.(Amati & Ounis, 2000)

2.7.1 Extended ontology:

The formal definition of Ontology based on the Conceptual Graph theory, provides a knowledge framework for ontology that is represented in the form of the type and relation hierarchy and a specific conceptual graph that represents the Ontology information. This specific CG resembles the knowledge of domain including generic and specific concepts and their interrelation regarding to ontology's framework (type hierarchy).

In (Amati & Ounis, 2000) the restrictive format of the CGs is introduced that each concept type in it takes only one referent to represent the specific concept. This format of conceptual graph is used to provide possibility of deduction system over the conceptual graph.

In the specific CG we consider the specific concepts as the new concept types and place them into the type hierarchy. Assume that we have a generic type [City: *] and we are interested in representing the specific concepts in the ontology for this type, such as [Ankara] and [Istanbul].

We call this new hierarchy as the closure of hierarchy of concept types. This new hierarchy adds the individuals into the concept type hierarchy. One of the main advantages of this extension is the ability to define composite types for individuals without any conjunction between the types.

Conjunction between types leads to some difficulties in the representation, since it needs to define the compatible concept types and banned concept types.

In our approach each individual is considered as the independent concept type and should be added into the concept type hierarchy. If the individual has a composite concept type then it will belong to all of these concept types. For example if we have individual [Reza] that is an international student then we need an [international student] concept type in the concept hierarchy, but even if there is not such a concept type in the hierarchy we can still present [Reza] as a new concept that has two concept types of [student] and [International]

2.8 Ontology and context-aware computing

Ontology is used to represent the entities and their interrelationships in a specific domain. In definition of Ontology the description of an entity is called a class or a concept. Each concept has attributes that describe slots (also called properties or roll) of each class. The restriction on slots is defined in facets (also called role restrictions). In order to generate Ontology we should first define classes of Ontology, and then arrange them in a granular hierarchy, and then we should define slots with their accepted values, and finally fill the slots with values. Different relations between the defined classes are illustrated by links. These links or relations describe two types of relation: internal links that describe relation among classes and instances like instance-of or subclass-of.(Noy et al., 2001)

In general most of Ontology languages are logic-based languages in order to define concepts, their properties and relationships in accurate, consistent and meaningful way (Heflin, 2004). Symbolic knowledge representation and description languages are both based on logic-based formalism. Description logic is a favorable representation that supports full reasoning that can be performed automatically. These abilities are widely used in OWL-DL as a solution for challenging sides of expressiveness and complexity of reasoning.(Bettini et al., 2010)

In general ontologies are exploit representation and reasoning abilities of logics in order to:

1. Describe complex context based on expressiveness of logic languages.
2. To define a semantic for concepts and their relations. This facilitates the integrating context or sharing it between various sources.

3. To use available tools for consistency checking and simple reasoning processes. (Bettini et al., 2010)

Context-aware applications need proper context modeling to be generic and to be usable for various applications. Context sharing and common understanding of context is main driver for recent research in the field of context representation and reasoning algorithms (Strang & Linnhoff-Popien, 2004).

2.9 Ontology-based context matching and related works

Three eminent works, SOCAM, CoBrA and the Context Managing Framework (Baldauf et al., 2007) which use the Ontology-based context modeling are summarized in this section. Our purpose is to point out which dimensions of context are used and how Ontology can be used to describe/model them.

Both CoBrA and SOCAM are using the OWL, whereas Context Managing Framework uses the RDF. The CoBrA uses its own Ontology language that is an OWL based language but called COBRA-ONT.

The SOCAM (Service-Oriented Context-Aware Middleware) architecture is used for rapid prototyping context-aware applications (Gu, Pung, & Zhang, 2005b).The context in basic model is described in the form of "*predicate (subject, value)*" like "location (john, bathroom)" that means john is located in bathroom and "Temperature (kitchen, 120)" that means temperature of kitchen is 120º F. In the same way, by adding Boolean operations, more complex contexts are generated. In order to be scalable SOCAM divides the pervasive environment into a set of sub-domains and uses different ontologies for different environments. In addition to generalized ontologies, the specific ontologies, that are also known as domain specific ontologies are the main parts of Ontology knowledge base. In requested situations these two ontologies are bound together, for example when user is in home the home Ontology and general Ontology are joined together.

Basic contexts defined in the generalized Ontology include person, location, activity and computational entity. These concepts and related properties and relations are illustrated in Figure 4 (Gu, Pung, & Zhang, 2005b).

Different reasoning systems can connect to this knowledge base and perform reasoning tasks (Miraoui, Tadj, & Ben Amar, 2008). Interpreter uses various ontologies, this reduce the performance of the system but increase the reusability of the system (Gu, Pung, & Zhang, 2005b). The interpreter of SOCAM is developed with Jena2 (HP's Semantic Web Toolkit) ("Jena Semantic Web Framework," n.d.) that is a Java framework for developing semantic web applications.

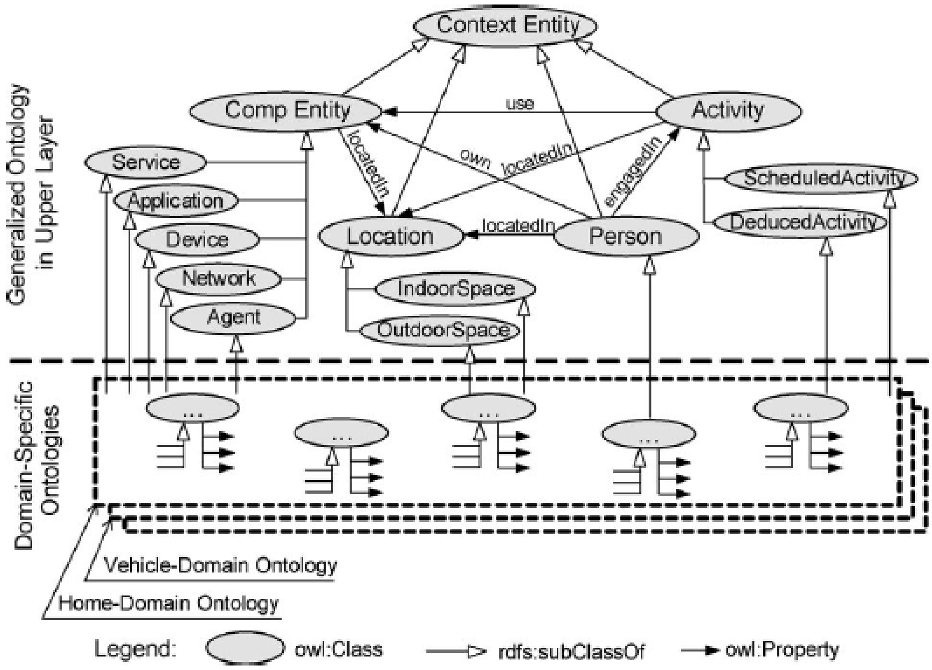

Figure 3 Generalized Ontology and relation with domain-specific Ontology (Gu, Pung, & Zhang, 2005b)

The Context Broker Architecture (in short form CoBrA) also benefits from the advantages of using Ontology. CoBrA as mentioned before uses the semantic web language OWL. OWL enables CoBrA to share information with environmental agents and in addition provides an Ontology model that is easy to reason and check the consistency of contextual information. OWL also facilitates sharing Ontology as a common knowledge between agents. There are four main categories of context in CoBrA: place, agent, agent's activity and agent's location (Chen et al., 2003) Ontology is also divided into four separate parts:

1. Ontologies about places: it categories concepts by defining physical places with their social specifications. For example, place refers to typical physical locations in a university campus

like campus, building, room. Spatial information like longitude and latitude are the properties of the place class.

2. Ontologies about agents: it categories concepts by defining software and human agents.
3. Ontologies about an agent's location context: it categories agent contexts based on their location. For example "PersonInBuilding" is a class that describes a group of people that are located in the building.
4. Ontologies about an agent's activity context: it categories concepts to describe the activity situation of agents.

Context Managing Framework (Korpipaa, Mantyjarvi, Kela, Keranen, & Malm, 2003) uses Ontology in order to define the valid contexts to be used by clients. Raw sensory data is converted into Ontology representation form. This representation of context has a higher reusability than sensory raw data and provides the ability to serve human interpretable context information for other applications. The Ontology consists of a schema that represents the structure and properties of concepts and the vocabulary that consist of terms in order to describe context information. The structure and properties are represented with RDF (Resource Description Framework) that enables the Ontology to be shared. Context expressions consist of six parameters, namely context type, context value, confidence, source, timestamp, and attribute. Context type and context value are mandatory information; an example for context type is "Environment:Sound:Hormonicity" and the value for this context type is "{Low, Medium, High}". Context type is represented the category of concept. Context's primitive categories are location, time, environment, user and device. Context value is used with the context type to describe the context. (Korpipaa & Mantyjarvi, 2003)

Dividing universal Ontology into different ontologies of specific domains is a pragmatic approach that is used in (Gu, Pung, & Zhang, 2005b).One of the advantages of using individual ontologies is the simplicity of their creation. The terms and relations in a specific domain are limited. In addition sentences that use the limited words of specific Ontology have similar structure (Zhong, Zhu, Li, & Yu, 2002).

2.10 Conceptual Graph matching

Gomez et al. (Montes-y-Gómez, López-López, & Gelbukh, 2000) used conceptual graph matching for textual information retrieval. In this work, text is represented in the form of CG and comparison is done on current and desired texts which correspond to the document and the user's query, respectively. The concept nodes of CG are used to represent the elements of text that includes nouns, verbs, adverbs, adjectives. The relation nodes of CG are used to represent some restricted relations between concepts like attributes prepositions, subject and object. For example, a sentences like "John loves Mary" is represented as a conceptual graph like [John]→(Subj.)→[Love]→(Obj.)→[Mary] where brackets illustrate concept nodes and parentheses show the relation nodes. A part-of-speech tagger, a syntactic parser, and a semantic analyzer are used to generate the conceptual graph of the sentences. The algorithm to compare two CGs in this work includes two main steps. The first step is to find intersection of two graphs. In the second step, the similarity of two graphs is calculated based on the intersection graph obtained in the first step.

The intersection graph is defined as $G_C = G_1 \cap G_2$. The concept nodes of this graph are all concept nodes that appear in both G_1 and G_2. The relation nodes of this graph are all relations that appear in graphs G_1 and G_2 . The similarity of two graphs is highly related to similarity of their concepts and relations. The similarity is found as $S=S_C*(a + b*S_R)$. Coefficients a and b are used to decrease the effect of relation similarity in the case of $S_R=0$. The values for a and b are selected as follows:

$$a = \frac{2\mathrm{n(Gc)}}{2\mathrm{n(Gc)} + \mathrm{mGc(G1)} + \mathrm{mGc(G2)}}$$

$$b = 1 - a$$

$n(G_C)$ is the number of concept nodes in G_C and $m_{Gc}(G_1)+m_{Gc}(G_2)$ is the total number of concepts in G_1 and G_2 that are connected to G_C nodes.

Conceptual similarity (S_c) and relational similarity (S_r) are calculated with the following equations:

$$\mathrm{Sc} = \frac{2\mathrm{n(Gc)}}{\mathrm{n(G1)} + \mathrm{n(G2)}}$$

Where $n(G_C)$ is the number of concepts in graph $G_C n(G_1)$ and $n(G_2)$ are the number of concepts in graphs G_1 and G_2, respectively.

$$S_R = \frac{2m(Gc)}{mGc(G1) + mGc(G2)}$$

Where $m(G_c)$ is the number of relation nodes in graph G_c and $mG_C(G_1)$ and $mG_C(G_2)$ are the numbers of relation nodes in immediate neighborhood of G_C nodes in both graphs G_1 and G_2.

In (Zhong et al., 2002) a semantic search approach based on conceptual graph theory is proposed. Semantic search supports searching for various categories of objects from hypertext to multimedia description. The proposed approach uses CGs in order to describe document content. Similarity between current context CG (called resource CG) and desired context CG (called query CG) represents semantic similarity between resource and query contexts. The similarity between CGs is defined according to the concepts and relation similarities. In order to calculate the similarity between two concepts, distances between them are calculated as $d_C(C_1, C_2)$ and the similarity between two concepts is defined as:

$$Sim_C(C_1, C_2) = 1 - d_C(C_1, C_2)$$

Concept hierarchy of the CG helps us to calculate the distance between two concepts. In this method every concept node has a 'milestone' value that is calculated as :

$$milestone(n) = \frac{1/2}{k^{l(n)}}$$

Where l(n) is depth of the node in hierarchy and k is a factor larger than 1 that shows the rate of decrease in value along the hierarchy. The default value for k is equal to 2.

.

The milestone value of each node is used to calculate the distances between nodes according to their distances from the common closest node or CCP. Distance between two nodes is calculated as:

$d_C(C_1,C_2)=d_C(C_1,CCP)+d_C(C_2,CCP)$

$d_C(C,CCP)=$ milestone(CCP)-milestone(C)

Then the similarity between relations is:

$Sim_r(r_Q,r_R)=1\text{-}d_r(r_Q,r_R) \Rightarrow$ 1 if r_Q subsumes r_R , 0 if others

where r_Q is relation in query CG (desired CG) and r_R is relation of resource CG (current context CG).

CG similarity is defined by using the concept similarity and relation similarity. In a CG each concept has some connections with the relations and each relation is related to a concept that introduces a sub-graph under that concept. The similarity between two CGs is defined as the similarity between two concepts and their related sub-graphs. In order to distinguish between entities according to their importance, "weights" can be used. More important entities have higher weight in contrast to entities with less importance.

Similarity of two CGs, C_Q and C_R, is defined as:

$$SoG(C_Q,C_R)= W(C_Q,C).Sim_C(C_Q,C_R)+Max_{\text{for each combination}}\{\Sigma_j\ w(Q\ ,\ j).Sim_r(r^j_Q\ ,\ r^j_R).[SoG(C^{rjQ}_Q,C^{rjR}_R)]$$

Where

- C_Q is the entry of the query CG, C_R is the entry of resource CG.
- $SoG(C_Q,C_R)$ is the similarity between two CGs.
- r^j_Q and r^j_R represents the j^{th} relation that connects the entry C_Q and C_R concepts of query and resource CGs.
- C^{rjQ}_Q and C^{rjR}_R are entry nodes of the sub-graphs those are attached to the relations r^j_Q and r^j_R
- $W(C_Q,c)$ and $W(C_Q j)$ represent the weights of the entry and the jth relation association with the entry respectively.

In (Laudy & Ganascia, 2010) the authors proposed a compatibility testing method based on conceptual graphs. The compatibility testing comes as a prior level before information fusion. They use advantages of conceptual graph theory for illustrating current situation. Matching measure for two graphs was used to understand compatibility rate by comparing with a threshold that is assigned by a domain expert.

The similarity between two graphs is computed as:

$\text{Sim}_{\text{match}}(G_1,G_2)=\frac{\Sigma(c1,c2)\in \text{match Sim concept}(c1,c2)}{\min(|C1|,|C2|)}$

where

- C_1 and C_2 are the set of concepts of G_1 and G_2
- $|C_1|$ and $|C_2|$ are the count of concepts in G_1 and G_2
- c_1 and c_2 are concepts of G_1 and G_2

The similarity between two graphs is calculated according to concepts with maximum similarity. In different possible matching of concepts the concepts with maximize similarity is taken into account.

Similarity between two concepts is computed by comparing two concepts based on their conceptual types, their values and their neighborhood. Similarity between two concepts can be calculated as:

$\text{Sim}_{\text{concept}}(C_1,C_2) = p_1(t_1,t_2)\text{sim}_{\text{Type}}(t_1,t_2)*(\frac{p2(T1,T2)\text{simRef}(v1,v2)+p3\text{sim Rel}(c1,c2)-p4\text{dissRel}(c1,c2)}{p2+p3})$

where

- c_1 and c_2 are concept nodes
- t_1 and t_2 are conceptual types that belong to c_1 and c_2
- v_1 and v_2 are individual markers that represent t_1 and t_2 values.
- p_1, p_2 , p_3 and p_4 are weights to emphasize on elements of higher importance.
- Sim_{Type} : similarity between conceptual types defines as the distances between them in lattice of concept. In order to find distance between two concepts
  - $\forall(t_1, t_2)\in S\times S$, $\text{dist}(t_1,t_2)= \min(t \leq t_1, t\leq t_2)$ $(\text{lS}(t_1,t_2)+\text{lS}(t_2,t))$
  - With $\forall(t_1, t_2)\in S\times S$, $t\leq t'$ $\text{lS}(t1, t')=\Sigma\ ti \in(t, t'), ti\neq t\ [\frac{1}{2\text{prof}(ti)}]$
- Sim_{Ref} : similarity between two referents
- Sim_{Rel} : similarity regarding to neighborhoods , comparing direct neighborhoods of concepts is proposed as a factor to find similarity between two concepts which is computed as:

  $\text{Sim}_{\text{Rel}}(c_1,c_2)=\frac{2*\text{nbRelComm}(c1,c2)}{\text{nbRel}(c1)+\text{nbRel}(c2)}$

Where $\text{nbRelComm}(c_1,c_2)$ is the number of relations that are common in c_1 and c_2 , and $\text{nbRel}(c_1)$ and $\text{nbRel}(c_2)$ are total number of relation connection to c_1 and c_2 nodes.

- $diss_{Rel}$: dissimilarity regarding neighborhoods. Comparing neighborhoods dissimilarity is another factor to find similarity between two concepts. It is computed as:

$$diss_{Rel}(c_1,c_2)=\frac{2*nbRelDiff(c1,c2)}{nbRel(c1)+nbRel(c2)}$$

nbRelD iff(c_1,c_2) shows the number of relations that are not common between c_1 and c_2

CHAPTER 3

CONTEXT MATCHING BASED ON CONCEPTUAL GRAPHS

In this thesis, we use conceptual graphs to model the current and desired contexts. Whenever the current context changes, the current and desired contexts are matched. The matching process consists of five main steps as illustrated in Figure 4. The basic idea employed in the matching process is to construct a CG corresponding to the current context. This graph is structured according to the incoming query so that the matching is facilitated. For the same purpose, we also preprocess the query CG to refine requested context by using ontology.

The first step in the matching process is the preparation step that includes generating the ontology and the observation units which are tiny conceptual graphs that represent main context dimensions including hour, date, status and location of a specific actor. The actor may be a person or a thing. In this step, the knowledge base of the domain will be generated and saved.

The second step is about converting the query CG into the desired context CG. The original query may come in the natural language and it is first represented by a conceptual graph. The conversion of natural language description of query into a conceptual graph is outside the scope of this study. In this thesis, we concentrate on converting the query CG to a form that can be processed easily, and this form is called desired context CG.

The context observation units that belong to different actors are generated in first step should be filtered according to the requested actor's context in the desired context CG. In this step among a multitude of the context units, the context observations that may be related to the desired context CG will be filtered. Then, these context units will be used to generate the current context CG.

In the following step, the filtered context units will be joined together according to the desired context CG relations. The generated new CG is the called the current context CG.

The final step computes the matching score for the desired context CGs and the current context CG.

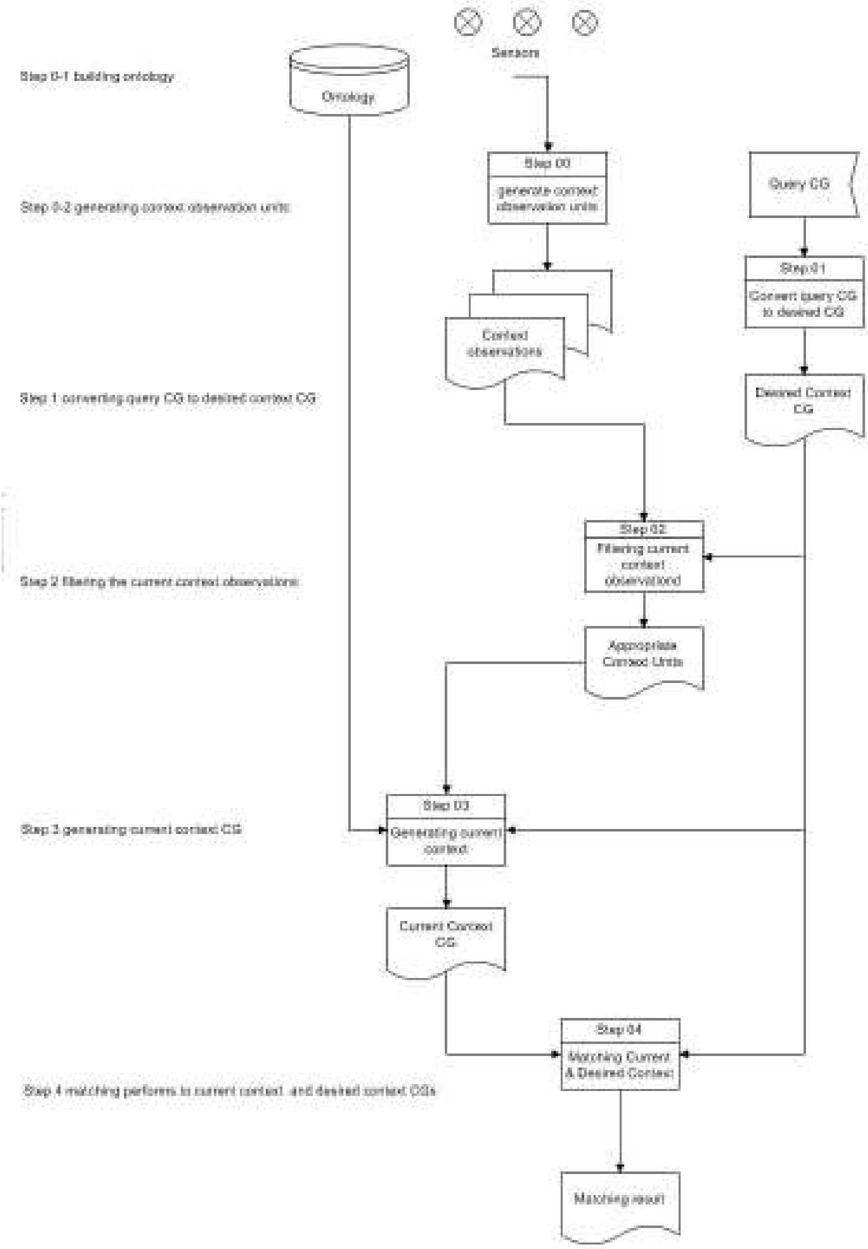

Figure 4 The context matching process

3.1 Step 1: Generating Ontology and current context observation units:

3.1.1 Generating ontology

In this thesis, we represent the Ontology by using the conceptual graphs. Our Ontology consists of a concept hierarchy, relation hierarchy and a specific graph that represent the domain knowledge by representing generic concepts, individuals and their interrelations.

We utilize the concept type and relation type hierarchy to compute concept type and relational type similarity between concepts and relations of desired context and current context CGs.

We form our Ontology based on five primitive categories, namely Location, Status and Person, Hour and Date. Parts of a sample Ontology for concept type and relation type hierarchies and sample domain knowledge are given in Figure 5, Figure 6, and Figure 7.

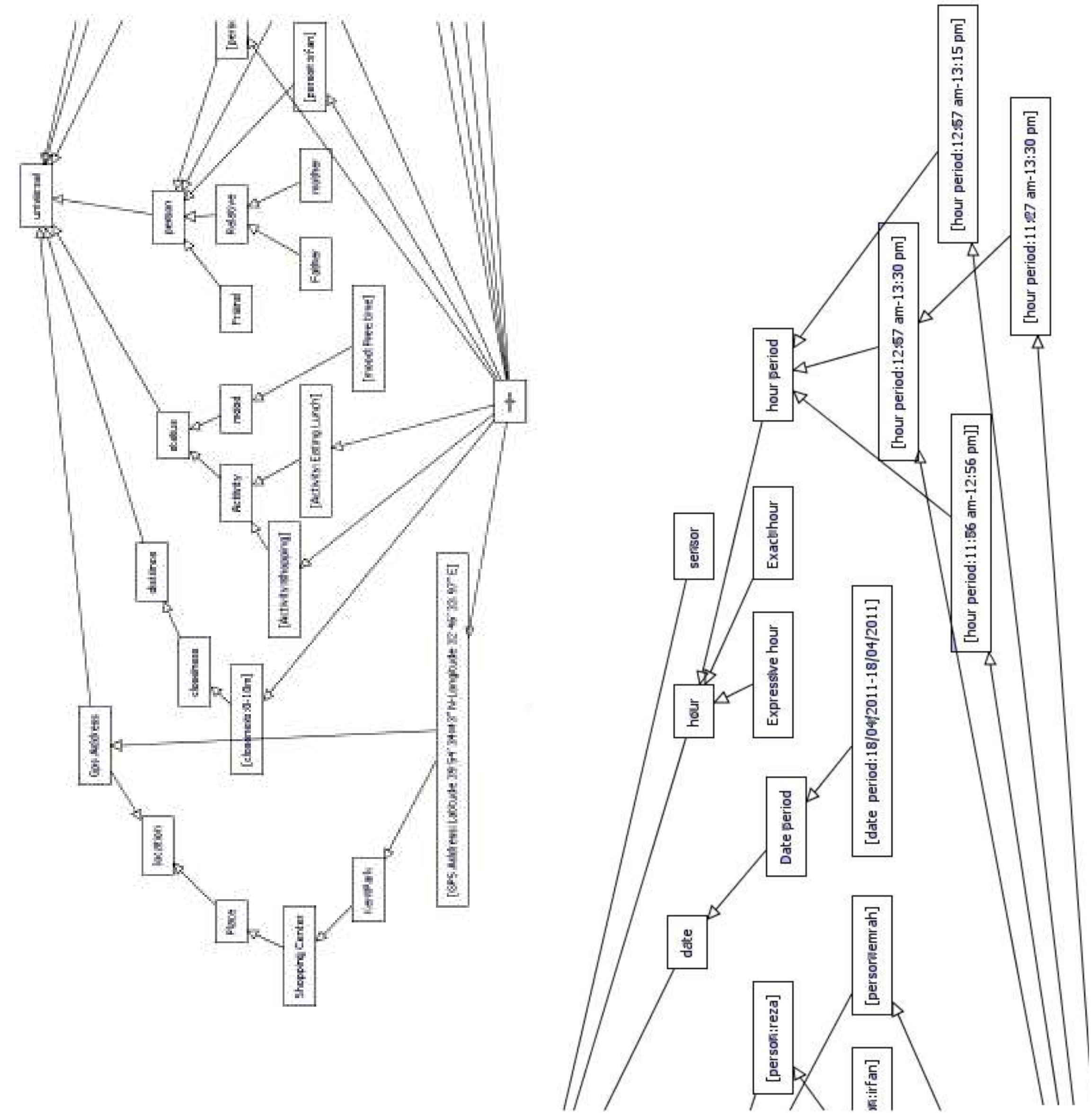

Figure 5 Sample concept type hierarchy in Ontology

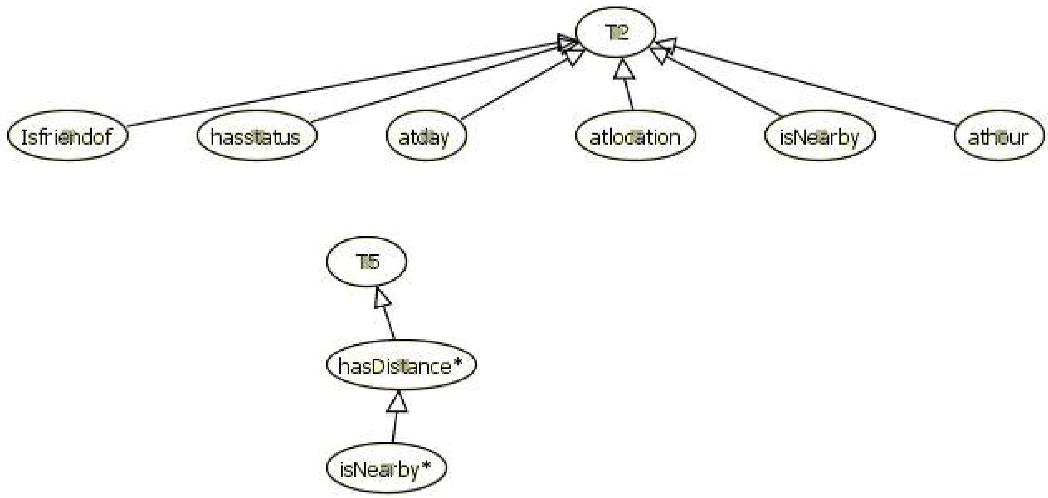

Figure 6 Sample relation type hierarchy in Ontology

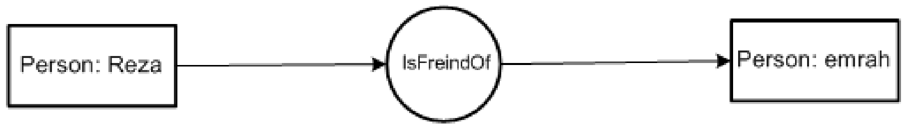

Figure 7 Sample information in Ontology

3.1.2 Generating current context observation units from sensory raw data:

Captured data from each sensor should be represented as tiny conceptual graph that we call context atom. A context atom is a CG that consists of a two concepts and a relation. The relation reflects the type of the sensor and related concepts represent the actor and sensed value. The actor may be a person or thing to which is the sensory data is associated.

Each context observation that belongs to an actor should be associated to hour and the date of the observation. As the sensing operation is continuous, instead of representing each sensed data together with date of observation separately, we create a single graph for a sensed data and associate duration of validity. That is, the time (hour and date) of each context unit is associated to a time period that starts form “start time” and ends in the ”end time”. The start time is set to when the context unit is generated and end time is updated as data comes from the related sensors. If the status and location of the actor in the context unit remains the same with the prior status and location, we only update the end time. If status and/or location changes, after updating the end time of the existing context unit, a new context unit for same actor with new values is generated. Tracing the status and location contexts are directly related to our interest, however other contexts may also be added to the observation units as needed.

"Start time" and "End time" of each context unit is kept in the form of the time range for both hour and date contexts. The Start time and End time in context unit refer to duration of context so it is reasonable to represent them as a range as the following:

[Hour: m] and [Date: n] with $d_{start}:h_{start} < n:m < d_{end}: h_{end}$

We use the main idea of (Pawar & Tokmakoff, 2006) to associate the actors to the sensors and a system employing our approach should support:

1. Registration of sensors dedicated to the actors which exist in incoming queries.
2. Capturing data from registered sensors.
3. Associating data captured from sensors to the related actors and the system clock.

As a summary, in order to construct context units, the following steps are taken:

```
1.   The context unit of registered actors is constructed with the location and status
        context atoms.
2.   The hour and date of captured context are set as the start hour and start date of the
        current context observation unit.
3. System periodically polls the registered sensors
 If there is no change in the context {
                Update the End hour and date.
      }
     If there are changes in status or location {
1- The End hour and date of the CG should be set to current hour and date.
2- The context unit is stored in context history.
3- New observation CG should be generated for this actor and start hour and date and
        end hour and date are set to current hour and date.
}
```

3.1.3 An example for representing sensory data as context units:

In Figure 8, a sample context observation unit is given. This unit contains the contextual information for the user [person:reza] in certain hour period [11:56-12:56] in specific location that is a GPS measurement on [18/04/2011] having the [Activity:eating lunch] status.

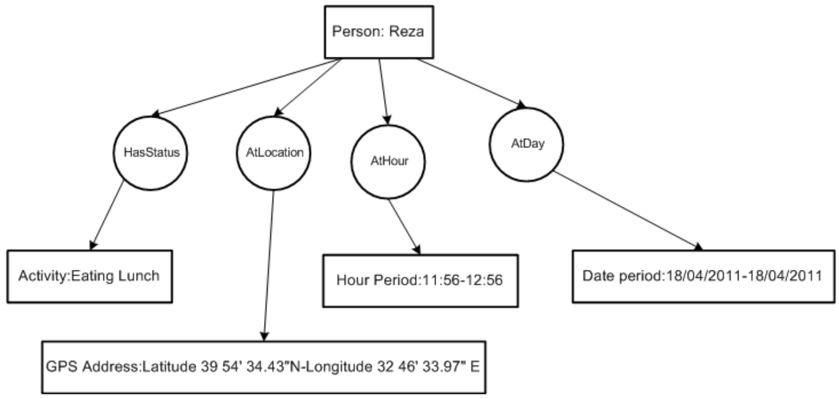

Figure 8 context unit of the actor REZA at 18/04/2011- 12:56 pm

As explained above, whenever status and location context information changes a new context unit will be generated. The context unit in Figure 9 reflects a change in the actor's status. As it can be seen end time of previous context unit and the start time of the new context unit are successive.

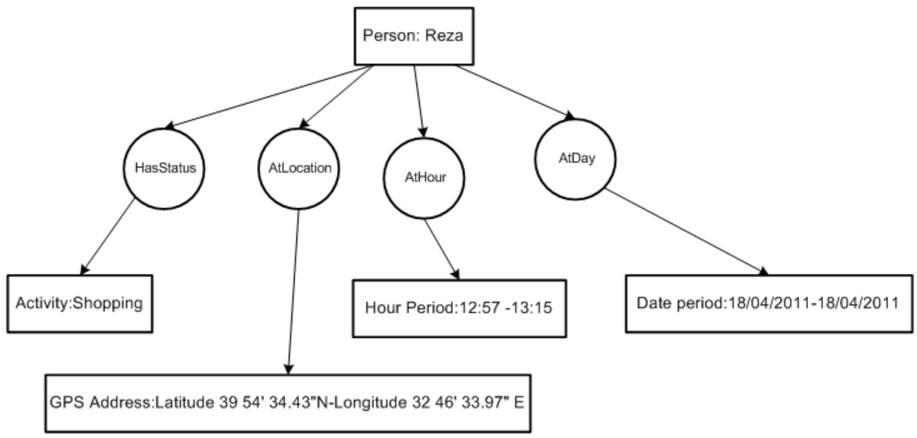

Figure 9 Context unit of actor Reza at the 18/04/2011-at: 12:57 pm -13:15 pm

By acquiring an observation that has the same information about the status and location of the actor (reza) the context unit in Figure 9 should be updated. Updating a context unit changes its end date/hour to current time of observation. The updated context unit is given in Figure 10.

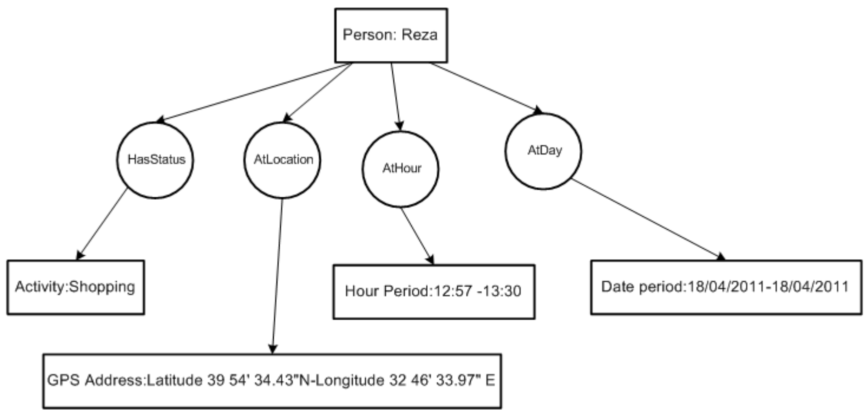

Figure 10 context unit for actor Reza at the 18/04/2011 at 13:30 pm

3.2 Step 2: Converting the query CG into desired context CG:

In this step, Query CG is converted to desired context CG by using the Ontology. The conversion algorithm is as follows:

1. Replace all existential referents into their equivalences, using Ontology.
2. Complete context units for each actor (context units are CGs that only has one actor and represent four main context dimensions such as date, hour, status and location).
3. Transliterate the complex relations into a collection of generic relations which has the same meaning with the complex relation.
4. Translate descriptive referents into the numeric values, using Ontology.

3.2.1 Transliteration of complex relations

In this step a complex relation in query CG is decomposed into some fundamental relations. We refer to a relation as complex when no instances with given arguments exist in the Ontology. Some complex relations resemble a reasoning result and should be extended to contain all related arguments to lead to the same result. It should be extended in the way to include fundamental context dimensions that are required for the reasoning. In other words, we extend the complex relation in order to have required arguments.

We use the main idea of “**Closure Ontology**” (P. Nguyen et al., 2008) that was proposed an extension on relations in the ontology to support new relations based on the current relations that require extra constraints or arguments in the relation.

We describe the relation extension as following: assume these relations offend (offender, offender victim) and steal (thief). We have steal≤offend that means offend is more generic than thief. As the query we have “did John steal from Mary? “. In order to represent this relation we extend the steal (thief) to a new relation steal*(thief, thief victim). While introducing new argument into existing relation we should be careful as selected argument should be a specific form of its corresponding argument in the generic relation. In this example, we should have thief<offender and thiefVictim≤offenderVictim in the concept type hierarchy. Adding new concept types into the concept type hierarchy requires reviewing the relations in the ontology, and we should be careful to prevent adding new concept types of minor signification. (P. Nguyen et al., 2008)

Extending a relation generates a more generic relation. Using the extended relation instead the complex relation has no side effect on matching results. The extended relation is considered as the generalization of the complex relation. The complex relation and the extended one have the maximum similarity and we are allowed to replace the complex relation with the extended one.

3.2.2 An example for Converting the query CG into desired context CG:

In this section, we illustrate the query CG to desired context CG conversion by an example. Consider the following incoming query as “Reza is nearby a friend”. The generated query CG based in this query is similar to following CG in Figure 11.

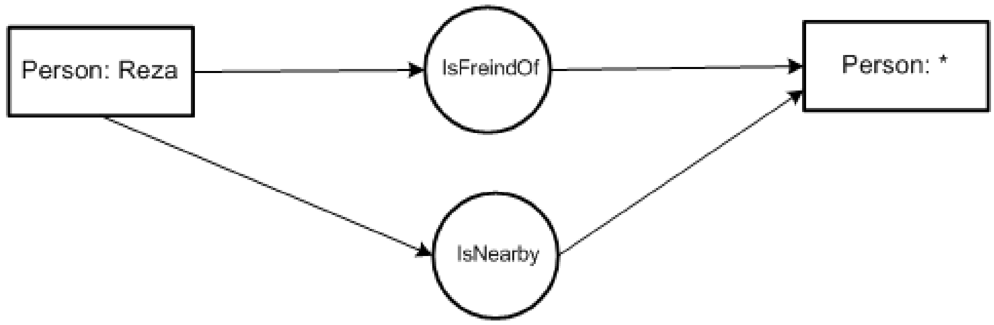

Figure 11 incoming query CG

1- All of the existential concepts should be interpreted according to Ontology. In this example, we have the generic concept [people:*] that satisfy the relation [person:reza]®(isFriendOf)®[people:*]. Based on the ontology information we have [person:reza]®(isFriendOf)®[people: Emrah]

The updated incoming query is presented in Figure 12.

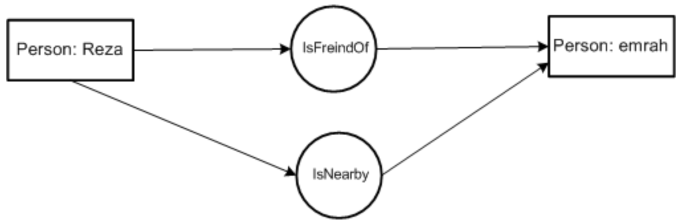

Figure 12 generic concepts converted the specific ones.

2- In second step we add missing dimensions to the actors in the query. If no specific values are specified for the main context dimensions then we should set their values as the generic concepts. The updated query CG is given in Figure 13. The hour and date context dimensions in this step are specified as the generic types, which means [hour period: current hour] and [date period : current date]

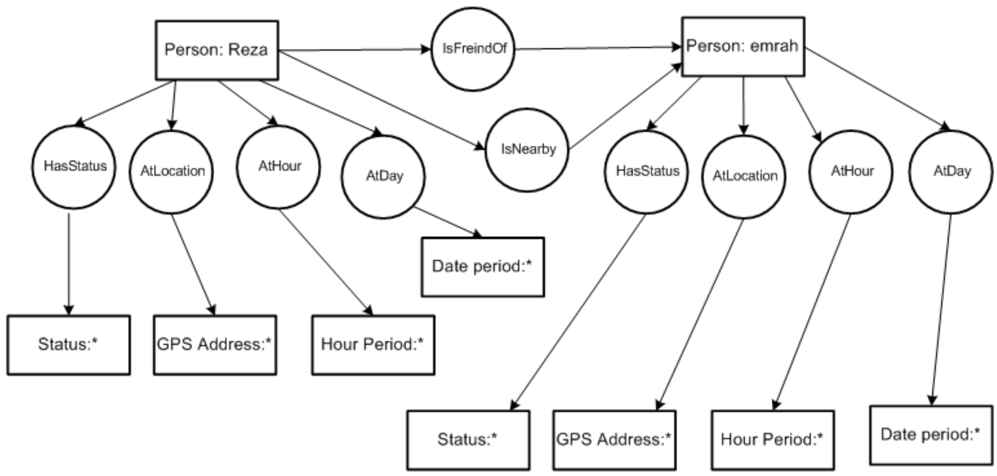

Figure 13 Query CG after adding the missing context dimensions into the actors,

3. The relation of isNearby<person:reza, person:*> is a complex relation, as we can't find it in the Ontology. In order to translate this complex relation, we refer to the relation type hierarchy.

Therefore isNearby(person,person) relation is found to be a specific type of the hasDistance(person,person,distance) relation. That is, isNearby≤hasDistance. The hasDistance relation extension is the hasDisatance*($person_1$,$person_2$, distance,$location_1$,$location_2$). Between these three relations we have isNearby≤hasDistance≤hasDistance*. The extension of the isNearby is defined as isNearby*(person,person,clossness). The closeness is defined as a new concept type according to the concept hierarchy as closeness≤distance. We also define the closeness between two persons as distance between persons' location context as a range of [0-10m]. We can extend it more to contain the person's location such as isNearby*($person_1$,person-$_2$,clossness, $location_1$, $location_2$). Therefore, we represent the "reza is nearby a person" as:

hasDistance*(person:reza,person:*,Distance:*,location:A,location,B)

IsNearby*(person:reza, person:*,closeness:[0-10m], location: A, location :B)

The resultant query CG is given in Figure 14.

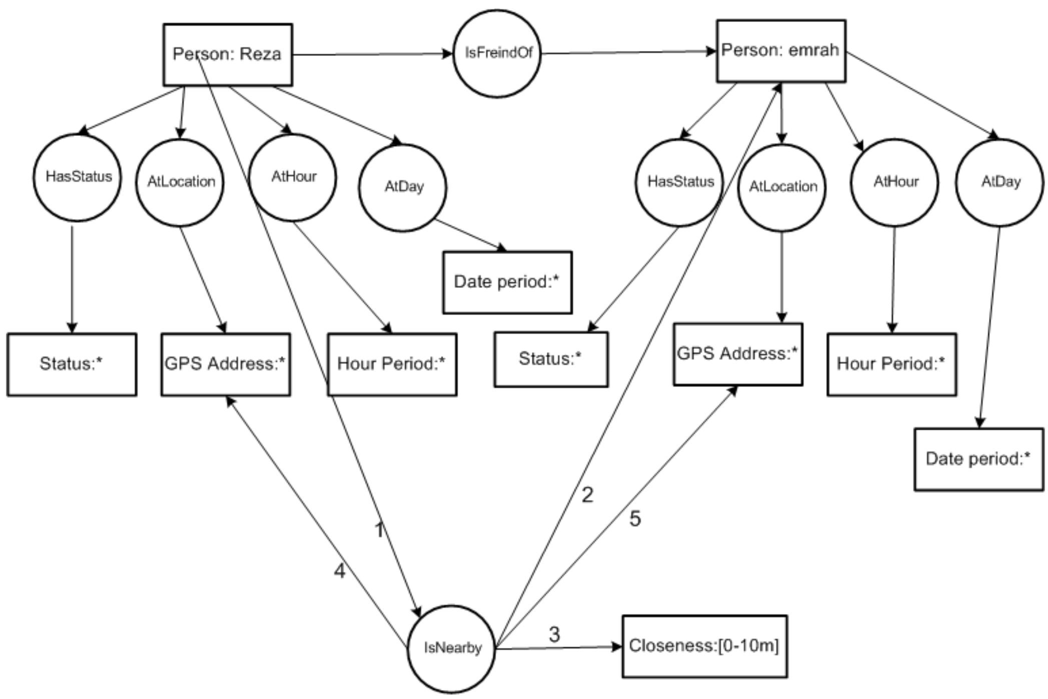

Figure 14 desired context CG

3.3 Step 3: Filtering the current context observation units

Context observation unit consists of context atoms which belong to the same actor at the same hour and date. Due to the rapid changes in the context information and large number of actors there may be a huge number of context units in a simple application scenario.

In this section we describe an algorithm to filter appropriate context units to speed up matching operation. The resultant context units will be used in the current context CG.

Filtering of context units is performed according to following steps:

1. We compare context units with the desired context CG in order to select appropriate context units that may be used in current context CG.
2. In the history search more than one result may be found for several actors. In this case, those units with different actors that occur in the same time should be considered together in order to generate current context CG.

3.3.1 An example for filtering and augmenting context units:

Suppose that the current hour is 13:30 pm and the current date as 18/04/2011.Current hour and date can be used to filter the context units which belong to Reza and Emrah (desired context CG actors). The selected context unit for actor Reza is given in Figure15 and the context unit of the actor Emrah is given in Figure 16. These context units are chosen because they have the highest similarity score in the matching with the desired context CG in contrast with the other context units.

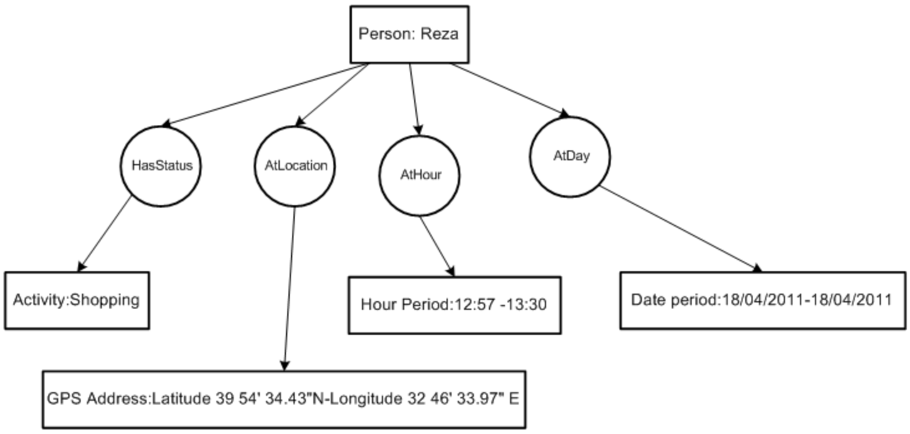

Figure 15 current context unit for Reza

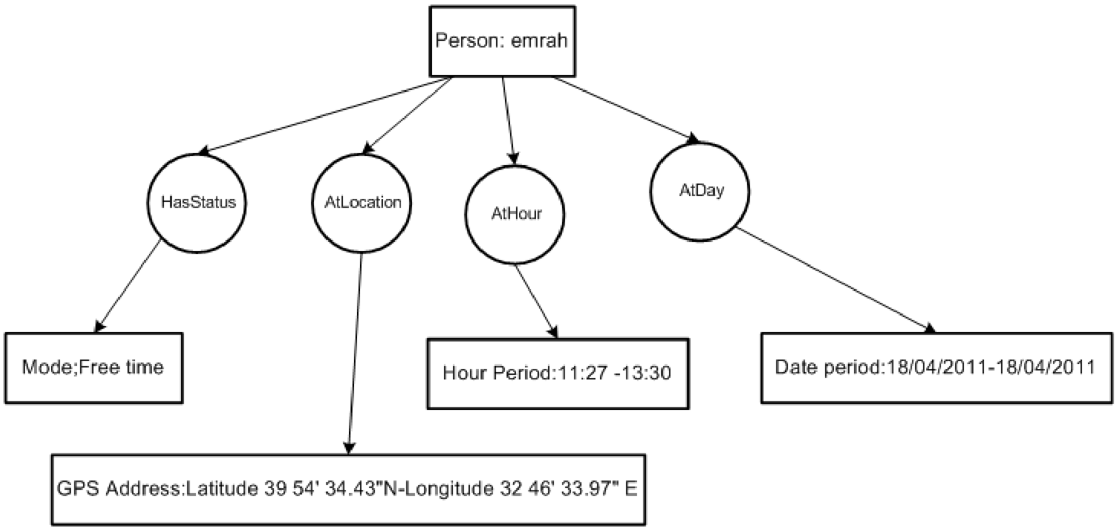

Figure 16 current context unit for Emrah

3.4 Step 4: Generating current context CG:

We make use of the structure of the desired context CG (Figure 14) in order to construct the current context CG. The desired context CG is the query CG that is augmented in order to replace the generic concepts with those individuals that have the most specific meaning according to Ontology. The steps for constructing the current context CG is:

1-Make a copy of the desired context CG.

2-replace the context units of each actor with the selected current context observation units.
3-compute the values for those arguments that are depending on hour, date, location and status.

3.4.1 An example of generating Current context CG

We generate the current context CG based on the desired context CG given in Figure 14 and the context units that are selected from the various context observations which are given in Figure 15 and Figure 16. In the current context CG we only replace these context units with the corresponding context units in the desired context CG. The generated current context CG is given in Figure 17.

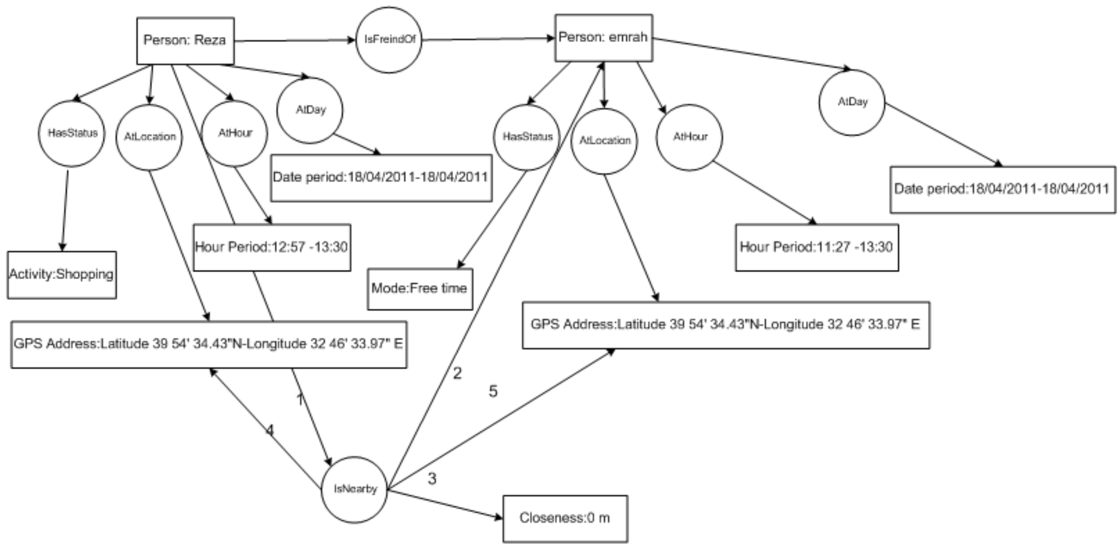

Figure 17 current context CG

3.5 Step 5: Matching between current context CG and desired context CG:

In this step the desired context CG and the current context CG is matched. The context matching approach is based on the conceptual graph matching. In (J. Sowa, 1984),(Foo, Garner, Rao, & Tsui, 1992),(Ralescu & Fadlalla, 1990) the similarity between the concepts and in (Zhong et al., 2002) and (Laudy & Ganascia, 2010) the similarity between CGs have been studied. We use the original idea in these references and extend it by the use of Ontology to support our purpose.

We define similarity between two conceptual graphs as the average of their relations' similarity. In our model, the relation similarity depends on the relation type similarity and conceptual similarity and conceptual similarity is the similarity between the concepts of a relation. We define the concepts as tuple of concept type and referent ([concept type: referent]). Concept type similarity multiplied by the referent similarity is defined as the conceptual similarity between two concepts and for all concepts in one relation the conceptual similarity is defined as the average of arguments similarity. Regarding to the nature of query situation some weights can be defined for each relation in desired context to discriminate more important context dimensions from less important ones. Each assigned weight should be between 0 and 1 and sum of the weights should be 1.

According to our model, similarity between two CGs D and C is computed as:

$$Sim_{CG}(D , C) = AVG(\Sigma \text{ Max } \{ Sim_{\text{relation type}} (r_{di} , r_{cj})* Sim_{\text{conceptual}} (r_{di} , r_{cj})\})$$
$$\text{for } 1\leq i\leq m \text{ and } 1\leq j\leq n$$

and if we use Weights then:

$$Sim_{CG}(D , C) = \Sigma w_i * \text{ Max } \{ Sim_{\text{relation type}} (r_{di} , r_{cj})* Sim_{\text{conceptual}} (r_{di} , r_{cj})\}$$
$$\text{for } 1\leq i\leq m \text{ and } 1\leq j\leq n$$

$1\leq i\leq m$ m=number of relations in D

$1\leq j\leq n$ n= number of relations in C

$1\leq s\leq p$ p=Arity of desired relation

$1\leq t\leq q$ q=Arity of current relation

Where,

- D refers to Desired context CG
- C refers to the current context CG
- w_i denotes the weight assigned to r_{di} from the desired CG
- r_{di} is the i-th relation in desired context CG , $r_{di} < c^1{}_{di},\ldots, c^s{}_{di,\ldots} , c^p{}_{di} >$ n=Arity of r_{di}
- r_{cj} is the j-th relation in current context CG, $r_{cj} < c^1{}_{cj},\ldots, c^t{}_{cj,\ldots} , c^q{}_{cj} >$ n=Arity of r_{cj}

In the above computation, we select and compare those relations from desired context CG and current context CG that have the maximum similarity. That is, we compare the selected relation from desired context CG with all relations in current context CG, one by one. The relation with highest similarity score will be selected from current context CG.

Conceptual similarity is defined between two relations and depends on relations' arguments similarity. Relation's argument or concepts are defined as a combination of concept type and referent such as $c^{s}_{di} = [tc^{s}_{di}: m^{s}_{di}]$. We take both concept type and referent similarity into account and calculate the conceptual similarity as:

$$Sim_{conceptual}(r_{di}, r_{cj}) = AVG\,(Sim_{Concept\ type}(tc^{s}_{di}, tc^{t}_{cj}) * Sim_{Referent}(m^{s}_{di}, m^{t}_{cj}))$$

for $i=1\ldots m$ and $j=1\ldots n$ and for $s=1\ldots p$ and $t=1\ldots q$

$r_{di} = <c^{1}_{di}, c^{2}_{di}, \ldots, c^{s}_{di}, \ldots c^{p}_{di}>$

$r_{cj} = <c^{1}_{cj}, c^{2}_{cj}, \ldots, c^{t}_{cj}, \ldots c^{q}_{cj}>$

$r_d = \{r_{d1}|r_{d2}|\ldots|r_{di}|\ldots|r_{dn}\}$

$r_c = \{r_{c1}|r_{c2}|\ldots|r_{cj}|\ldots|r_{cm}\}$

m= number of relations in D, n= number of relations in C

3.5.1 Relation Type Similarity:

We defined relation type similarity measures how much two relation's relation type is similar together as:

$$Sim_{relation\ type}(r_{di}, r_{cj}) = 1 \quad \text{If } r_{cj} \le r_{di}$$
$$= 0 \quad \text{Otherwise the similarity value is 0.}$$

We give the maximum similarity between two relation types if one relation type subsumes the other one in the relation type hierarchy in the Ontology. Otherwise the similarity is set to zero and further attempts to find the similarity between their concepts makes no sense.

As an example, we compute the relation type similarity for the desired context CG in Figure 14 and the current context CG in Figure 18 and represent those pairs with highest similarity as following are:

1. $Sim_{relation\ type}$ (atlocation1, atlocation1)=1
2. $Sim_{relation\ type}$ (hasStatus1, hasStatus1)= 1,

3. $Sim_{relation\ type}$ (Athour1, Athour1)=1,
4. $Sim_{relation\ type}$ (Atday1, Atday1)=1 ,
5. $Sim_{relation\ type}$ (atlocation1, atlocation2)=1
6. $Sim_{relation\ type}$ (hasStatus1, hasStatus2)= 1,
7. $Sim_{relation\ type}$ (Athour1, Athour2)=1,
8. $Sim_{relation\ type}$ (Atday1, Atday2)=1 ,
9. $Sim_{relation\ type}$ (atlocation2, atlocation1)=1
10. $Sim_{relation\ type}$ (hasStatus2, hasStatus1)= 1,
11. $Sim_{relation\ type}$ (Athour2, Athour1)=1,
12. $Sim_{relation\ type}$ (Atday2, Atday1)=1 ,
13. $Sim_{relation\ type}$ (atlocation2, atlocation2)=1
14. $Sim_{relation\ type}$ (hasStatus2, hasStatus2)= 1,
15. $Sim_{relation\ type}$ (Athour2, Athour2)=1,
16. $Sim_{relation\ type}$ (Atday2, Atday2)=1 ,
17. $Sim_{relation\ type}$ (isFriend , isFriend)=1 ,
18. $Sim_{relation\ type}$ (isNearby* ,isNearby*)=1 ,
19. $Sim_{relation\ type}$ (Distance , Distance)=1 ,

3.5.2 Concept Type Similarity:

Each concept in conceptual graph consists of two parts the concept type and the referent. The similarity between The concept type of two concepts such as [tc^s_{di} : m^s_{di}] and [tc^s_{cj} : m^s_{cj}] is defined based on the concept type hierarchy of Ontology. We use extended concept type hierarchy in order to keep individuals as new concept types. The concept type similarity between two concepts is computed as:

$$Sim_{concept\ type}(tc^s_{di}, tc^t_{cj}) = \begin{cases} 1 & \text{If } tc^t_{cj} \leq tc^s_{di} \\ 0.50 & \text{If } tc^s_{di} \leq tc^t_{cj} \\ 0 & \text{If } CCP = \top \\ 1-Diss_{concept\ type}(tc^s_{di}, tc^t_{cj}) & \text{Otherwise} \end{cases}$$

- tc^s_{di} is the concept type of s-th argument of relation r_{di}, equivalently [tc^s_{di} : m^s_{di}]

- tc^{t}_{cj} is the concept type of s-th argument of relation r_{cj}, equivalently $[tc^{t}_{cj} : m^{t}_{cj}]$

In order to compute the dissimilarity between two concept types or $Diss_{type}(t_i, t_j)$ we should compute the distance between them. In the concept type hierarchy (T_C) , we find the common closest parent (CCP) of t_i and t_j . The distance between two concept types are defined as the total of their distances from the CCP (Common Closest Parent). Path selection does not have an impact on the distance value because scores are only depends on the depth of node.

$$Diss_{type}(t_i, t_j) = Distance\ (t_i, ccp) + Distance\ (t_j, ccp)$$

$$Distance\ (t, ccp) = milestone\ (ccp) - milestone(t)$$

Milestone is used to find the depth of the concept node. Milestone is computed as:

$Milestone\ (t) = ½ / 2^{level(t)}$ level(t) shows depth of node.

In some special situations there is no need to calculate the distance between concepts in the concept type hierarchy explicitly. Here we give some special situations:

1- We have CCP of two concepts as $\top$ then :
   The concept similarity between two concepts set to zero.
2- The desired concept type is more generic that the current concept type or $t_j \leq t_i$ then the CCP will be same as the t_i that is a concept in desired context CG and t_j as the sub-type of the t_i .The similarity between a super-type and sub-type is in highest rate and we should set the similarity value to 1.
3- The desired concept type is more specific than current concept type or $t_i \leq t_j$ .In this case we assign 0.5 as the concept type similarity score.
4- If we have the t_i and t_j as direct sub-types of the CCP, that means $Milestone(CCP)=Milestone(t_i)+Milestone(t_j)$

We check the concept type similarity between the relations in desired context CG in Figure 12 and the current context CG in Figure 16. For all relation combinations we check their concept type similarity as:

1. $Sim_{relation\ type}$ (atlocation1, atlocation1)=1

$Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (GPS address,GPS address)=1

2. $Sim_{relation\ type}$ (atlocation1, atlocation2)=1

   $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (GPS address,GPS address)=1
3. $Sim_{relation\ type}$ (atlocation2, atlocation1)=1

   $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (GPS address,GPS address)=1
4. $Sim_{relation\ type}$ (atlocation2, atlocation2)=1

   $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (GPS address,GPS address)=1
5. $Sim_{relation\ type}$ (hasStatus1, hasStatus1)= 1,

   $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (activity,activity)=1
6. $Sim_{relation\ type}$ (hasStatus1, hasStatus2)= 1,

   $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (activity,mood)=0.75
7. $Sim_{relation\ type}$ (hasStatus2, hasStatus1)= 1,

   $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (mood,activity)=0.75
8. $Sim_{relation\ type}$ (hasStatus2, hasStatus2)= 1,

   $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (mood,mood)=1
9. $Sim_{relation\ type}$ (Athour1, Athour1)=1,

   $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (Hour period, Hour period)=1
10. $Sim_{relation\ type}$ (Athour1, Athour2)=1,

    $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (Hour period, Hour period)=1
11. $Sim_{relation\ type}$ (Athour2, Athour1)=1,

    $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (Hour period, Hour period)=1
12. $Sim_{relation\ type}$ (Athour2, Athour2)=1,

    $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (Hour period, Hour period)=1
13. $Sim_{relation\ type}$ (Atday1, Atday1)=1 ,

    $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (Date period, Date period)=1
14. $Sim_{relation\ type}$ (Atday1, Atday2)=1 ,

    $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (Date period, Date period)=1
15. $Sim_{relation\ type}$ (Atday2, Atday1)=1 ,

    $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (Date period, Date period)=1
16. $Sim_{relation\ type}$ (Atday2, Atday2)=1 ,

    $Sim_{concept\ type}$ (person,person)=1 $Sim_{concept\ type}$ (Date period, Date period)=1
17. $Sim_{relation\ type}$ (isFriend , isFriend)=1 ,

$Sim_{concept\ type}(person, person)=1$ $Sim_{concept\ type}(person, person)=1$

18. $Sim_{relation\ type}(isNearby^*, isNearby^*)=1$,

$Sim_{concept\ type}(person_1, person_1)=1$, $Sim_{concept\ type}(person_2, person_2)=1$
$Sim_{concept\ type}(closeness, closeness)=1$
$Sim_{concept\ type}(GPS\ address_1, GPS\ address_1)=1$,
$Sim_{concept\ type}(GPS\ address_2, GPS\ address_2)=1$

3.5.3 Referent Similarity

We mentioned that concepts in the conceptual graph represented in form of a combination of concept type and the referent. Referent also may be considered as the referent's value. In order to find the similarity between two concepts computing the referent similarity is necessary. Referents may obtain numeric or non-numeric values.

We define the maximum similarity between a generic concept in the desired context CG and an individual in the current context CG. For all individual we have m $\leq$* that means the generic symbol subsumes all individuals. It means the generic concepts in desired context CG and any individuals have the maximum referent similarity. The similarity between referents is defined as:

$$Sim_{Referent}(m_{di}, m_{cj}) = \begin{cases} 1 & \text{if } m_{di}=m_{cj} \\ 1 & \text{if } m_{di} \neq m_{cj} \text{ and } m_{di}=* \text{ or } m_{cj} \leq m_{di} \\ Sim_{relational}(m_{di}, m_{cj}) & \text{otherwise} \end{cases}$$

For non numeric referents we have

$$Sim_{relational}(m_{di}, m_{cj}) = Sim\ sub_{CG}(C_{di}, C_{cj})$$

For Numeric referents we calculate their similarity based on their distance from each other

$$Sim_{relational}(m_{di}, m_{cj}) = Sim_{value}(Distance(m_{di}, m_{cj}))$$

3.5.3.1 Referent similarity in non-numeric referents:

The similarity between two non-numeric referents is defined as their relational similarity. Our main interest is to propose a generic method for the people and status contexts that always take non-numerical values. For those contexts which take non-numeric values we choose to translate those non-numeric values into the numeric ones by using the Ontology.

We define the referent similarity for people and status contexts' individuals as their neighborhood similarity. With the neighborhood we mean the sub-graphs that are representing the main contextual dimensions in the current context CG and desired context CG. The main common context dimensions that we consider for people's neighborhood are status, location, hour and date and for the status context we consider are location, hour and date. The main idea is that people normally behave the same at specific locations at specific times. For example what people do during the lunch break in a normal work day at cafeteria? The status of each user may be set to different things but depending on its type similarity and the relational similarity we represent their status similarity. Same thing happens for people context, the relational similarity of people who are in same location at same hour and date with the similar status at the maximum level. The relational similarity is supposed as categorization for the non-numeric individuals. Those individuals in the same category are more similar in contrast to other individuals out of the category.

In order to compute the relational similarity between two individuals we compute the similarity between their first level sub-graphs. We use modified version of conceptual graph matching algorithm for this task. That is, we perform the matching only for first level sub-graphs. The algorithm is:

Set weight scores for each relation

Set the $Sim_{referent}$(Entrance D, Entrance C)=0

For each relation in Sub-graph D and relations in Sub-graph C

{

Calculate the similarity between each relation pair

Select relation pairs with maximum similarity score

For each selected relations $_{1\leq i\leq m}$

{

For each relation argument $_{2\leq s\leq p}$ // we omit the first argument

{

Calculate the average of concepts similarity

$Sim_{conceptual}(r_{di}, r_{cj})=AVG_{2\leq s\leq p}\,(Sim_{concept\ type}(tc^{s}_{di}, tc^{t}_{cj}) * Sim_{Referent}(m^{s}_{di}, m^{t}_{cj}))$

Calculate the MAX{ Sim $_{relation\ type}(r_{di}, r_{cj})$* Sim $_{conceptual}(r_{di}, r_{cj})$}

{

}

Compute the similarity score as :

Sim sub$_{CG}$(D , C) = Σ w_i* Max { Sim $_{relation}(r_{di}, r_{cj})$* Sim $_{conceptual}(r_{di}, r_{cj})$}

Return similarity score value

The conceptual similarity between two different status (or people) depends on their concept type similarity and their relational similarity. Suppose that two different statuses are [Activity: Shopping] and [Mode: Free time]. The neighborhoods of these statuses are:

[activity:shopping]®
-(AtHour)®[hour period:12:57 pm-13:30 pm]
-(AtDay)®[date period:18/04/2011-18/04/2011]
-(AtLocation)®[GPS Address: **Latitude 39 54’ 34.43” N-Longitude 32 46’ 33.97” E**]

[mode:free time]®
-(AtHour)®[hour period:11:27 pm- 13:30 pm]
-(AtDay)®[date period:18/04/2011-18/04/2011]
-(AtLocation)®[GPS Address: **Latitude 39 54’ 34.43” N-Longitude 32 46’ 33.97” E**]

The result of computing relational similarity between [Activity: Shopping] and [mode: Free time] will be:

Sim $_{relation\ type}$(AtHour, AtHour)*(Sim $_{concept\ type}$ (Hour Period,Hour Period)*Sim $_{referent}$(12:57 pm-13:30 pm, 11:27 pm-13:30 pm)= 1*1*0.65= 0.65

Sim $_{referent}$(12:57 pm-13:30 pm, 11:27 pm-13:30 pm)= 0.65

Range$_{sim}$(D,C)= (d_2-d_1)= 2:03

Range$_{diss}$(D,C)=(d_1-c_1)+ (c_2-d_2)=1:30

Diss (D,C)=|Range$_{sim}$(D,C)-Range$_{diss}$(D,C)| / Range$_{lenght}$(D)+Range$_{lenght}$(C)

=(2:03-1:30)/(2:03+00:33)=33/186=0.35

Sim (D, C)=1-Diss(D,C) =1-0.35=0.65

Sim $_{relation\ type}$(AtDay, AtDay)*(Sim $_{concept\ type}$ (date period,date period)*Sim $_{referent}$(18/04/2011-18/04/2011, 18/04/2011-18/04/2011)= 1*1*1=1
Sim $_{relation\ type}$(AtLocation,AtLocation)*Sim $_{concept\ type}$ (GPS Address, GPS Address)*Sim $_{referent-}$(**Latitude 39 54' 34.43" N-Longitude 32 46' 33.97" E, Latitude 39 54' 34.43" N-Longitude 32 46' 33.97" E**)=1*1*1=1

The relational similarity between [activity: shopping], [mode: free time] is calculated as:

Sim $_{relational}$ (shopping, free time)=AVG(0.65+1+1)=0.88.

In order to calculate the conceptual similarity between [activity: shopping] and [mode: free time] , we should first calculate their concept type similarity based on the concept type hierarchy that was given in Figure 5,we have Sim $_{concept\ type}$(activity, mode)=0.625 and for conceptual similarity based on its definition Sim $_{conceptual}$(C_1,C_2)=Sim $_{concept\ type}$(tc_1,tc_2)*Sim $_{referent}$(m_1,m_2), we have Sim $_{conceptual}$(C_1,C_2)=Sim $_{concept\ type}$(Activity,mode)*Sim $_{relational}$ (shopping, free time)

= 0.625*0.88= 0.55

As another example suppose the relational similarity for people context below:

[Person: Reza]® -(Atlocation)®[GPSAddress: **Latitude 39 54' 34.43" N-Longitude 32 46' 33.97" E**] -(HasStatus) ®[activity:shopping] -(AtHour)®[hour period:12:57 pm- 13:30 pm] -(AtDay)®[date period:18/04/2011-18/04/2011] [Person: Emrah]® -(Atlocation)®[GPSAddress: **Latitude 39 54' 34.43" N-Longitude 32 46' 33.97" E**] -(HasStatus) ®[mode:Free time] -(AtHour)®[hour period:11:27 pm- 13:30 pm] -(AtDay)®[date period:18/04/2011-18/04/2011]

In the following the relational similarity's computation is detailed:

Sim $_{relation\ type}$(AtLocation, AtLocation)*(Sim $_{concept\ type}$ (GPS address, GPS address)*Sim $_{referent-}$

(**Latitude 39 54' 34.43" N-Longitude 32 46' 33.97" E**, **Latitude 39 54' 34.43" N-Longitude 32 46' 33.97" E**)=1*1*1=1 $Sim_{relation\ type}$(HasStatus, HasStatus)*($Sim_{concept\ type}$ (activity, mode)*$Sim_{referent}$(Shopping, free time)= 1*0.625*0.88=0.55 $Sim_{relation\ type}$(AtHour, AtHour)*($Sim_{concept\ type}$ (Hour Period,Hour Period)*$Sim_{referent}$(12:57 pm-13:30 pm, 11:27 pm-13:30 pm)=1*1*0.65=0.65 $Sim_{relation\ type}$(AtDay, AtDay)*($Sim_{concept\ type}$ (Date Period,Date Period)*$Sim_{referent}$(18/04/2011-18/04/2011, 18/04/2011-18/04/2011)= 1*1*1=1

$Sim_{Relational}$ ([person:reza],[person:emrah])=AVG(1,0.55,0.65,1)=0.8

$Sim_{Conceptual}$([person:reza],[person:emrah])=

$Sim_{concept\ type}$ (person,person)*$Sim_{Relational}$ ([person:reza],[person:emrah])=1*0.8=0.8

In summary, for the given desired context CG (Figure 14) and current context CG (Figure 17) we have two non-numeric references with the referent similarity as following:

$Sim_{Referent}$(reza,reza)= 1 $Sim_{Referent}$(reza,emrah)=0.8 $Sim_{Referent}$(emrah,emrah)=1 $Sim_{Referent}$(emrah,reza)=0.8	$Sim_{Referent}$(*,shoppig)= 1 $Sim_{Referent}$(*,free time)=1

This approach can be used to check the referent similarity between a numeric referent and a non-numeric referent.

3.5.3.2 Referent similarity for numeric scores:

For numeric referents the difference between their values can be used to find the dissimilarity between them. Computing distance between two numeric referents is a bit tricky depending on whether we have an exact value or a range of numeric data. We consider each combination differently as explained in the following:

1- **Desired concept's referent and current concept's referent have range values**: In the case of having range values for current and desired context: We represent the desired context range as range D with start point d_1 end points d_2. For current context we assign range C with start point c_1 and end points c_2. Dissimilarity between two ranges will be calculated as following:

Diss (D,C)=|$Range_{sim}$(D,C)-$Range_{diss}$(D,C)| / $Range_{lenght}$(D)+$Range_{lenght}$(C)
Sim (D, C)=1-Diss(D,C)
$Range_{lenght}$(D)=d_2-d_1 and $Range_{lenght}$(C)=c_2-c_1

We calculate the $Range_{sim}$(D,C) and $Range_{diss}$(D,C) based on the situation.The possible relations between ranges D and C are as following:

$d_1 = c_1$ & $d_2 = c_2$	sim(D,C)=1
$d_1<c_1<c_2<d_2$	sim(D,C)=1
$c_1<d_1<d_2<c_2$	$Range_{sim}$(D,C)= (d_2-d_1) $Range_{diss}$(D,C)=(d_1-c_1)+ (c_2-d_2)
$d_1<c_1<d_2<c_2$ $c_1<d_1<c_2<d_2$	$Range_{sim}$(D,C)=(|c_1+d_1|) $Range_{diss}$(D,C)= (|d_1-c_1|)+(|c_2-c_1|)

2- **Desired concept's referent has range value and current concept's referent has certain value:** If the desired concept's value is in range (d_1,d_2) but the current concept's value is a certain value (c) the possible cases are:

$d_1 \leq c \leq d_2$	Sim(D,c)=1
$c <d_1$ & $c<d_2$ $c>d_1$& $c >d_2$	Sim(D,c)=0

3- **Desired concept's referent has certain value and current concept's referent has range value:** In this case desired context take a certain value (d) and current context is specified as a range C, the possible cases are:

$c_1 \leq d \leq c_2$	Sim(d,C)=1
$d<c_1$ & $d<c_2$ $d>c_1$& $d >c_2$	Diss(d,C)=min{|c_1-d|,|d-c_2|} / Range(c_2-c_1) Sim(d,C)=1-Diss(d,C)

4- **Desired concept's referent and current concept's referent have certain values:** If both desired and current context take certain values then the similarity of these referents will be computed as:

$Sim_{Referent}(d,c)=1-Diss_{Referent}(d,c)$
$Diss_{Referent}(d,c)$ mapped value to Distance(d ,c)=|d-c| from related table

Similarity scores for these values are set based on concepts' value difference and related similarity score. These scores are available in lookup tables, which are generated by the rule of thumb by domain experts. We present the similarity mapping table for location in Table 2, hour in Table 3 and date in Table 4. According to the information which is represented in Table 2, 3, 4 it is clear that less distance between contexts returns more similarity score.

In Table 2, the similarity scores for numeric location contexts are given. We use this table to calculate the similarity between two locations with numeric referents given as the GPS position information, we measure their distance and based on it we assign similarity value from Table 2. With some examples we try to illustrate how to compute similarity between two locations in GPS position values as follows:

Assume two places named KENTPARK and CEPA with certain GPS positions as:

GPS position of CEPA= Latitude 39 54' 37.08" N , Longitude 32 46' 42.12" E
GPS position of KENTPARK= Latitude 39 54' 34.62" N , Longitude 32 46' 33.82" E
Distance (CEPA, KENTPARK) = 0.12 Km
$Sim_{Referent}$ (CEPA, KENTPARK)=0.7

The Distance (CEPA, KENTPARK) is computed with aid of a GPS distance calculator.

Table 2 similarity scores based on the distance

level	Difference	$Diff_{value}$	Sim_{value}
1	≥2.001 km	1	0
2	0.751km- 2 km	0.8	0.2
3	0.151km-0.750km	0.5	0.5
4	0.041km-0.150 km	0.3	0.7

5	0.021 km -0.040 km	0.15	0.85
6	0.006 km-0.020 km	0.05	0.95
7	0 km-0.005km	0	1

In Table 3 the similarity score for Hour contexts is represented. In order to calculate similarity score between hour contexts we calculate the difference between values and use the information in Table 3. Here we have some examples:

Sim_{value} (15:06, 16:00) =1-$Diss_{value}$ (15:06, 16:00) = 1-0.3 =0.7
Sim_{value} (15:06, 17:00) =1-$Diss_{value}$ (15:06, 17:00) =1-1= 0
Sim_{value} (15:06, 18:00) =1-$Diss_{value}$ (15:06, 18:00) = 1-1=0
Sim_{value} (15:06, 19:00) =1-$Diss_{value}$ (15:06, 19:00) = 1-1=0

Table 3 similarity score based on Hour difference

level	Difference	$Diff_{Value}$	Sim_{Value}
1	≥1:31	1	0
2	1:16-1:30	0.8	0.2
3	1:01-1:15	0.5	0.5
4	00:46-1:00	0.3	0.7
5	00:31-00:45	0.15	0.85
6	00:16-00:30	0.05	0.95
7	00:00-00:15	0	1

In Table 4 similarity scores for date context is represented. Referent similarity for date contexts with the numeric values will be calculated based on their difference and mapped to similarity score according to Table 4. The referent similarity scores for date contexts are also assigned by rule of thumb.

Referent similarity is in its maximum when both current date context and query date context are the same. For those date contexts that have one day difference we assign the referent similarity according to their hour referent similarity. In other words if $Diss_{value}$(date $_C$, date $_D$)=1 then Sim_{value} (date $_C$, date $_D$)= Sim_{value} (hour $_C$,hour $_D$). Some examples are given as following:

Assume the Desired context and Current context as

Desired context [date period: 05/04/2011-05/04/2011] and [Hour period: 23:00-23:00] Current context [date period: 06/04/2011-06/04/2011] and [Hour period: 00:01-00:01]

As it can be seen the date difference between the desired context and current context is $\text{Diss}_{\text{value}}$ (05/04/2011, 06/04/2011) = 1 then the referent similarity for hour will be used as the referent similarity for date context or $\text{Sim}_{\text{value}}(\text{date}_D, \text{date}_C)=\text{Sim}_{\text{value}}(\text{hour}_D, \text{hour}_C)$ that we have $\text{Sim}_{\text{value}}(23{:}00, 00{:}01)= 1- \text{Diss}_{\text{value}}(23{:}00, 00{:}01)= 1- 0.5 = 0.5$

This referent similarity score for Hour context will be used as the referent similarity for date context then $\text{Sim}_{\text{value}}$ (05/04/2011, 06/04/2011) =0.5

Table 4 similarity scores based on date difference

level	Day difference	$\text{Diff}_{\text{value}}$	$\text{Sim}_{\text{value}}$
1	2 days >=	1	0
2	1 day	$\text{Diff}_{\text{value}}(\text{hour}_{C.}, \text{hour}_D)$	$\text{Diff}_{\text{value}}(\text{hour}_{C.}, \text{hour}_D)$
3	0 day	0	1

3.5.4 Matching algorithm:

At the first step we set weights for relations in the desired context CG. Then we take one relation in desired context CG and compare it with the relations in current context CG, one by one. The relation with highest similarity score will be selected from current context CG. For these selected relations we check the conceptual similarity. The conceptual similarity between two relations is defined as the average of corresponding argument's conceptual similarity. The conceptual similarity between arguments is the concept type similarity multiplied by the concept value similarity.

After checking the conceptual similarity we check if these arguments are used in other relations' argument or not. In other words if these concepts have sub-graphs we have to continue to check the similarity for sub-graphs.

Checking similarity for sub-graphs is the same as checking similarity for graphs. It starts by comparing relations and then concepts. If there are other sub-graphs in the compared sub-graphs they are checked recursively.

These steps of the algorithm are:

For each relation in current context CG and relations in desired context CG

{

Calculate the similarity between each relation pair

Select relation pairs with maximum similarity score

For each selected relation $_{1 \le i \le m}$ $_{m=\text{Relation number in D}}$

{

    For each relation argument $_{1 \le s \le p}$ $_{p=\text{Arity of current relation}}$

    {

        Calculate the relation's conceptual similarity

        $Sim_{conceptual}(r_{di}, r_{cj}) = AVG\,(Sim_{concept\ type}(tc^{s}_{di}, tc^{t}_{cj}) * Sim_{Referent}(m^{s}_{di}, m^{t}_{cj}))$

        Calculate the $MAX\{ Sim_{relation\ type}(r_{di}, r_{cj}) * Sim_{conceptual}(r_{di}, r_{cj})\}$

            {

                For the equal values select the most significant one.

            }

    }

}

Compute the graph similarity

$$Sim_{CG}(D, C) = \underset{\text{for } 1 \le i \le m \text{ and } 1 \le j \le n}{AVG(\, Max\, \{ Sim_{relation}(r_i, r_j) * Sim_{conceptual}(r_{di}, r_{cj})\}\,)}$$

Return similarity score value

As an example, we apply our matching algorithm for matching the desired context CG in Figure 14 and the current context CG in Figure 18. The detailed computations are given below. Note that, in the following, we only present those relations with maximum similarity.

Note: In the following, we use K instead of the [Latitude 39 54' 34.43" N-Longitude 32 46' 33.97" E].

- r_{d1}= atLoccation<person: Reza, GPS Address: *>
- r_{c1}= atLocation<person: Reza, GPS Address: k >
- $Sim_{relation\ type}$(atlocation,atlocation)=1
- $Sim_{concept\ type}$ (person,person)=1
- $Sim_{Referent}$ (reza, reza)=1
- $Sim_{concept\ type}$ (GPS Address, GPS Address)=1
- $Sim_{Referent}$ (*, K)=1

$Sim_{relation\ type}$(atLocation, atLocation)* ((Sim $_{concept\ type}$ (person,person)*$Sim_{referent}$(reza,reza) + Sim $_{concept\ type}$ (GPS address, GPS address)*Sim $_{referent}$(*, k))/2=1

- r_{d2}=athour < person:reza, hour period: 13:30-13:30> // current hour:13:30
- r_{c2}=athour < person:reza ,hour period:12:57-13:30>
- $Sim_{relation\ type}$(athour,athour)=1
- $Sim_{concept\ type}$ (person, person)=1
- $Sim_{referent}$ (reza,reza)=1
- $Sim_{concept\ type}$ (hour period, hour period)=1
- $Sim_{referent}$ (13:30-13:30,12:57-13:30)=1

$Sim_{relation\ type}$(athour, athour)* ((Sim $_{concept\ type}$(person,person)*Sim-$_{referent}$(reza,reza) + Sim $_{concept\ type}$ (hour period, hour period)*Sim $_{referent}$(13:30-13:30, 12:57 -13:30))/2=1

- r_{d3}=atdate<person:reza, date period: 18/04/2011-18/04/2011>//current date
- r_{c3}=atdate<person:reza, date period:18/04/2011-18/04/2011>
- $Sim_{relation\ type}$(Atday, Atday)=1
- Sim $_{concept\ type}$(person,person)=1
- $Sim_{referent}$(reza,reza)=1
- Sim $_{concept\ type}$(date period, date period)=1
- Sim $_{referent}$ (18/04/2011-18/04/2011, 18/04/2011-18/04/2011)=1

$Sim_{relation\ type}$(atday, atday)* ((Sim $_{concept\ type}$(person,person)*Sim-$_{referent}$(reza,reza) + Sim $_{concept\ type}$ (date period, date period)*Sim $_{referent-}$(18/04/2011-18/04/2011, 18/04/2011-18/04/2011))/2=1

- r_{d4}=hasstatus<person:reza, status:*>
- r_{c4}=hasstatus<person:reza, activity:shopping>
- $Sim_{relation\ type}$(hasStatus, hasStatus)=1

- Sim $_{\text{concept type}}$(person,person)=1
- Sim$_{\text{referent}}$(reza,reza)=1
- Sim $_{\text{concept type}}$(Status, activity)=1
- Sim $_{\text{referent}}$ (*,shopping)=1

Sim$_{\text{relation type}}$(hasStatus, hasStatus)* ((Sim $_{\text{concept type}}$(person,person)*Sim-$_{\text{referent}}$(reza,reza) + Sim $_{\text{concept type}}$ (Status, Activity)*Sim $_{\text{referent}}$(*, shopping))/2=1

- r_{d5}=isNearby*<person:reza, person:Emrah,closeness:[0-10m], GPS Add:*,GPS Add:*>
- r_{c5}= isNearby*<person:reza, person:Emrah,closeness:[0m], GPS Add:k, GPS Add:k>
- Sim$_{\text{relation type}}$(isNearby*, isNearby*)=1
- Sim $_{\text{concept type}}$(person,person)=1
- Sim$_{\text{referent}}$(reza,reza)=1
- Sim $_{\text{concept type}}$(person,person)=1
- Sim$_{\text{referent}}$(emrah,emrah)=1
- Sim $_{\text{concept type}}$(closeness,closeness)=1
- Sim$_{\text{referent}}$([0-10m],[0m])=1
- Sim $_{\text{concept type}}$(GPS address,GPS address)=1
- Sim$_{\text{referent}}$(*,k)=1
- Sim $_{\text{concept type}}$(GPS address, GPS address)=1
- Sim$_{\text{referent}}$(*,k)=1

  Sim$_{\text{relation type}}$(isNearby*, isNearby*) * ((Sim $_{\text{concept type}}$(person,person)*Sim$_{\text{referent}}$(reza,reza) + Sim $_{\text{concept type}}$(person,person)*Sim$_{\text{referent}}$(emrah,emrah) + Sim $_{\text{concept type-}}$(closeness,closeness)*Sim$_{\text{referent}}$([0-10m],[0m]) + Sim $_{\text{concept type}}$ (GPS address, GPS address)*Sim $_{\text{referent}}$(*, k)+ Sim $_{\text{concept type}}$ (GPS address, GPS address)*Sim $_{\text{referent}}$(*, k))/5=1

- r_{d6}=isfriendof<person:reza, person:emrah>
- r_{c6}= isfriendof<person:reza, person:emrah>
- Sim$_{\text{relation type}}$(isfriendof, isfriendof)=1
- Sim $_{\text{concept type}}$(person,person)=1
- Sim$_{\text{referent}}$(reza,reza)=1
- Sim $_{\text{concept type}}$(person, person)=1
- Sim $_{\text{referent}}$ (emrah,emrah)=1

$Sim_{relation\ type}$(isfriendof, isfriendof)* ((Sim $_{concept\ type}$(person,person)*Sim-$_{referent}$(reza,reza) + Sim $_{concept\ type}$ (person, person)*Sim $_{referent}$(emrah, emrah))/2=1

- r_{d7}=athour < person:emrah, hour period:13:30-13:30> // current hour
- r_{c7}=athour < person:emrah ,hour period:11:27-13:30>
- $Sim_{relation\ type}$(athour,athour)=1
- Sim $_{concept\ type}$ (person, person)=1
- Sim $_{referent}$ (emrah,emrah)=1
- Sim $_{concept\ type}$ (hour period, hour period)=1
- Sim $_{referent}$ (13:30-13:30,11:27-13:30)=1

$Sim_{relation\ type}$(athour, athour)* ((Sim $_{concept\ type}$(person,person)*Sim-$_{referent}$(emrah,emrah) + Sim $_{concept\ type}$ (hour period, hour period)*Sim $_{referent}$(13:30-13:30, 11:27-13:30))/2=1

- r_{d8}=atdate<person:emrah, date period:18/04/2011-18/04/2011> //current date
- r_{c8}=atdate<person:emrah, date period:18/04/2011-18/04/2011>
- $Sim_{relation\ type}$(Atday, Atday)=1
- Sim $_{concept\ type}$(person,person)=1
- $Sim_{referent}$(emrah,emrah)=1
- Sim $_{concept\ type}$(date period, date period)=1
- Sim $_{referent}$ (18/04/2011-18/04/2011, 18/04/2011-18/04/2011)=1

  $Sim_{relation\ type}$(atday, atday)* ((Sim $_{concept\ type}$(person,person)*Sim-$_{referent}$(emrah,emrah) + Sim $_{concept\ type}$ (date period, date period)*Sim $_{referent-}$(18/04/2011-18/04/2011, 18/04/2011-18/04/2011))/2=1

  - r_{d9}=hasstatus<person:emrah, status:*>
  - r_{c9}=hasstatus<person:emrah, mood:free time>
  - $Sim_{relation\ type}$(hasStatus, hasStatus)=1
  - Sim $_{concept\ type}$(person,person)=1
  - $Sim_{referent}$(emrah,emrah)=1
  - Sim $_{concept\ type}$(Status, mode)=1
  - Sim $_{referent}$ (*,free time)=1

$Sim_{relation\ type}$(hasStatus, hasStatus)* ((Sim $_{concept\ type}$(person,person)*Sim-$_{referent}$(emrah,emrah) + Sim $_{concept\ type}$ (Status, mood)*Sim $_{referent}$(*, free time))/2=1

- r_{d10}= atloccation1<person: emrah, GPS Address: *>
- r_{c10}= atlocation1<person: emrah, GPS Address: k >
- Sim $_{\text{relation type}}$(atlocation, atlocation)=1
- Sim $_{\text{concept type}}$ (person,person)=1
- Sim $_{\text{Referent}}$ (emrah , emrah)=1
- Sim $_{\text{concept type}}$ (GPS Address, GPS Address)=1
- Sim $_{\text{Referent}}$ (*, K)=1

$Sim_{\text{relation type}}$(atLocation, atLocation)* ((Sim $_{\text{concept type}}$ (person,person)*Sim_{referent}(emrah,emrah) + Sim $_{\text{concept type}}$ (GPS address, GPS address)*Sim $_{\text{referent}}$(*, k))/2=1

$Sim_{CG}(D,C)=AVG(Sim_{relation}(r_{d1},r_{c1}), Sim_{relation}(r_{d2},r_{c2}), Sim_{relation}(r_{d3},r_{c3}), Sim_{relation}(r_{d4},r_{c4}), Sim_{relation}(r_{d5},r_{c5}), Sim_{relation}(r_{d6},r_{c6}), Sim_{relation}(r_{d7},r_{c7}), Sim_{relation}(r_{d8},r_{c8}), Sim_{relation}(r_{d9},r_{c9}), Sim_{relation}(r_{d10},r_{c10}))=1$

The similarity score is equal to 1 that shows that desired context (Figure 14) and the Current context (Figure 18) are matched with maximum similarity.

CHAPTER 4

SAMPLE USAGE SCENARIO

In this section we try to apply the proposed matching method to a specific domain of the context-aware applications, Cinema domain.

4.1 Cinema domain:

In the cinema domain we have different concepts such as Movie Theater, movies, and the customers. Different relations between these concepts are defined when the customers are interested in to be informed about the movie's subject and its start time. Suppose that the customers want to go to cinema with some friends in special time. Other information about the ticket price or special promotions in addition to some extra details about actors/actresses which have played in the movie and a brief story of the movie will help the customers to decide to watch or don't watch the movie. Some extra factors also might be considered such as friends who are like the movie or other friend's ratings and comments about the movie.

In our context matching model, all of these information and interrelations between concepts must be defined in the ontology. In this chapter we describe how our method for best matching based on the ontology can be applied to various scenarios in the cinema domain.

4.2 Step0-1: Building Ontology

The ontology is represented in the form of a conceptual graph. The conceptual graph consists of a support and graph itself. The support of the CG is mentioned as the framework of the ontology and the graph is considered as the populated ontology. The framework of ontology consists of the concept type hierarchy and relation type hierarchy.

The concept type hierarchy is shown in Figure 18. We will benefit from the concept hierarchy for defining the concept type similarity in the matching process.

We define different concept categories such as location, status, date, time, person and movie and we describe the individuals as new concept types in the concept hierarchy. For example in MovieCategory branch we have documentary, action, comedy, biography, adventure and thriller. All of these concept types in the classic view should be defined as individuals such as [movieCategory: documentary]. In our approach we keep the concept type embedded in the individuals by defining them as new concept types. The concept type similarity between the individuals is defined as their distance from the closest common parent, and all direct children of the parent (sibling nodes) have maximum concept type similarity. In order to define the zero similarity between some concepts we can use the absurd type in the hierarchy and connect the concepts to this type. Those concepts with absurd CCP have zero concept type similarity. We mentioned that it is not allowed to define the same concept in different branches.

Each individual that is defined in the concept hierarchy as a new concept type can have more than one parent. This definition allows us to take concept conjunction into the account. Consider two concept types namely [bird] and [fish-eater], the conjunction of these two types may be considered as the [fish-eater-bird], but without any conjunction and by defining [pelican] as the direct subtype of the [bird] and [fish-eater].

In the concept type hierarchy we have [Latitude 39 54' 34.43" N-Longitude 32 46' 33.97" E] which is the GPS address of the Kentpark. This is located under the [GPS Address] and [Kentpark]. The embedded meaning of this conceptual definition can be considered as "[Latitude 39 54' 34.43" N-Longitude 32 46' 33.97" E] is the GPS Address of Kentpark". This help us to determine [Reza]®(atPlace)®[Kentpark] and [Reza]®(atLocation)®[Latitude 39 54' 34.43" N-Longitude 32 46' 33.97" E] as two relations with maximum similarity. Based on relation type hierarchy we have atlocation<atplace that returns the $Sim_{relation\ type}$ (atplace,atlocation)=1 and according to the concept type hierarchy we have [Latitude 39 54' 34.43" N-Longitude 32 46' 33.97" E] ≤ [KentPark] so the concept type similarity of this GPS address and Kentpark are 1.

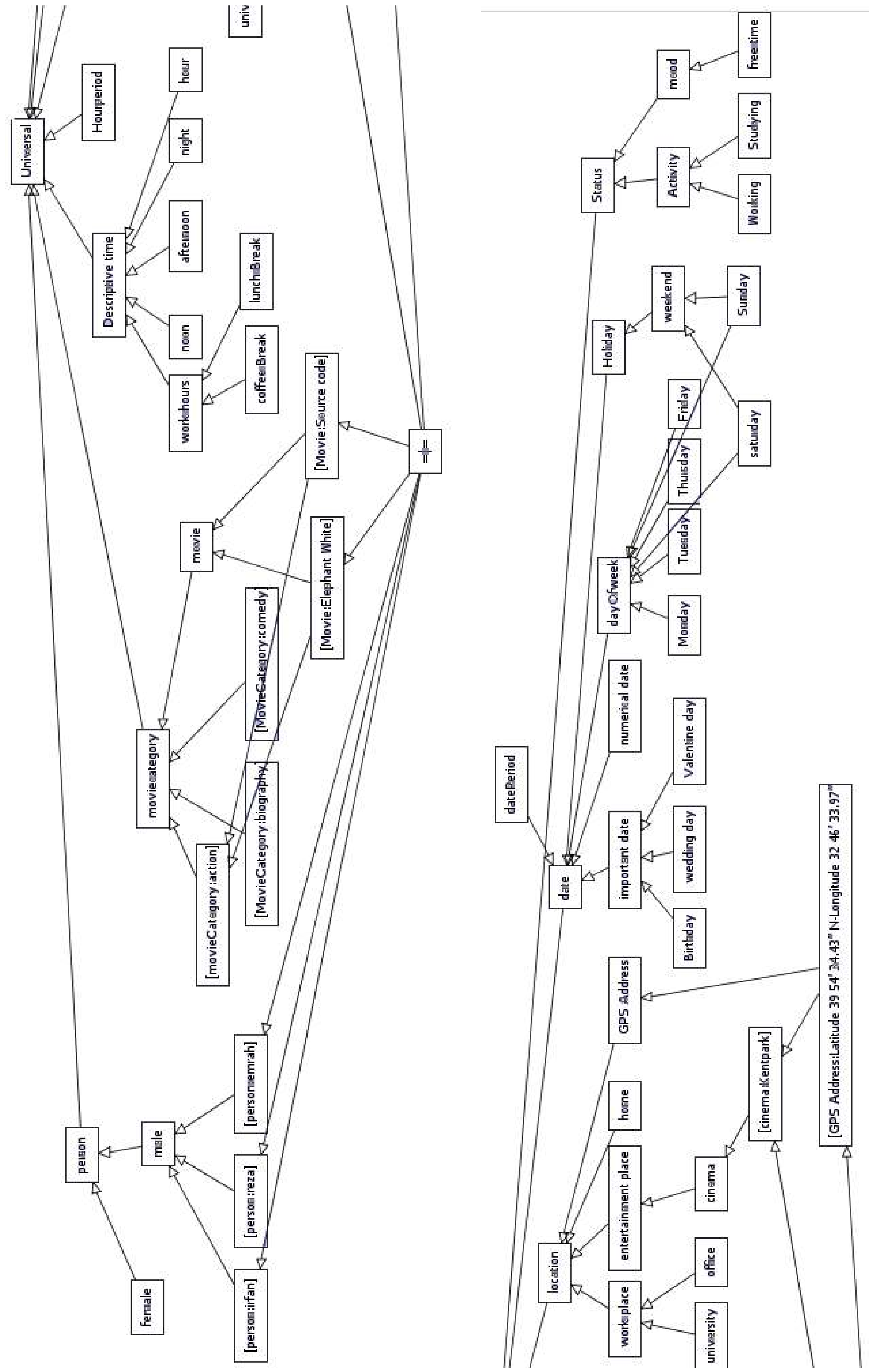

Figure 18 concept type hierarchy of Ontology

The relation type hierarchy of the ontology is given in Figure 19. Relational hierarchy is important to define the relation type similarity which is an important factor in computing similarity score between two conceptual graphs and extending relations.

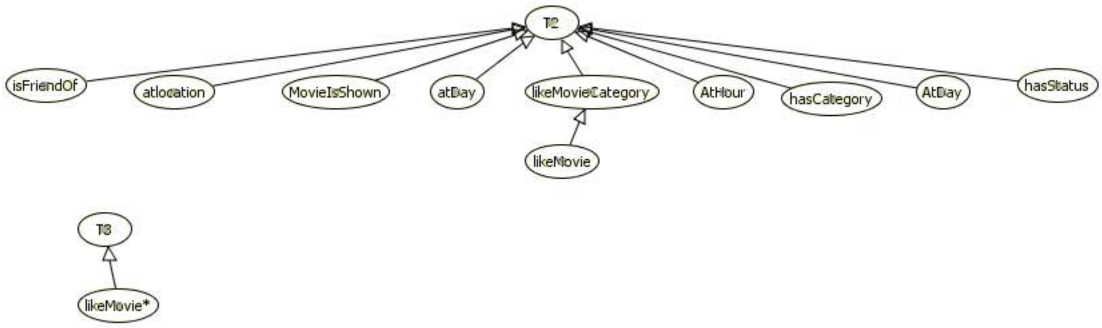

Figure 19 Relation type hierarchy of Ontology

The concept type hierarchy and relation type hierarchy which are mentioned as Ontology framework are also considered as unpopulated Ontology. A specific conceptual graph that is generated based on this framework is considered as the populated Ontology which consists of the concepts and interrelations. A sample Ontology is given in Figure 21.

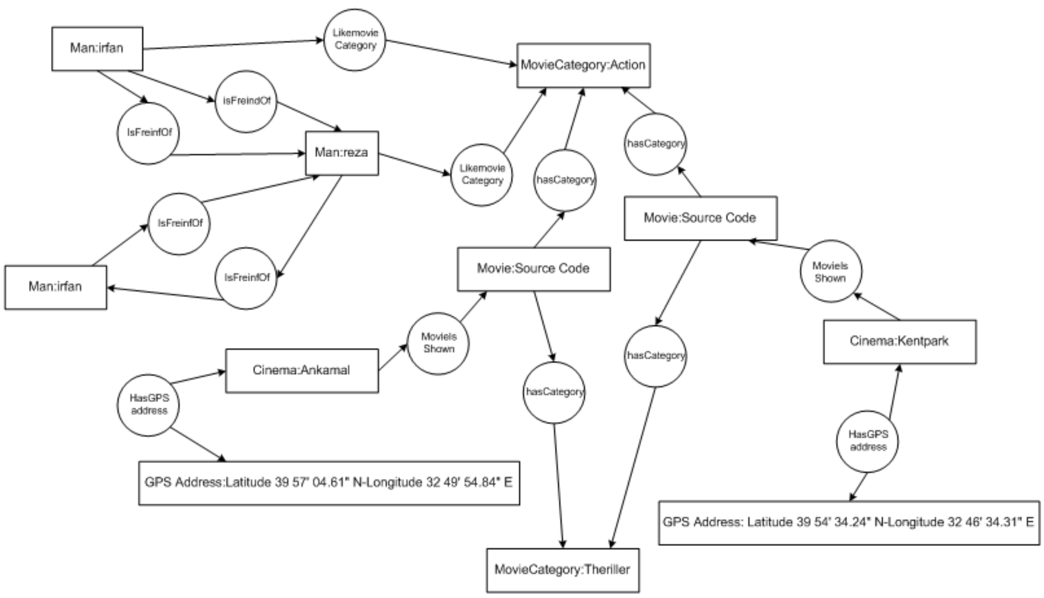

Figure 20 Populated Ontology of Cinema Domain

In the ontology we have Irfan as a person who is a friend of Reza and also likes action movies. We have Reza who is friend of Irfan and Emrah and likes action movies. We have Emrah who is friend of Reza. We have two movies in the ontology named “elephant white” and “source code” both in action and thriller category and the early one is shown in Ankamall and the last one is shown in Kentpark cinemas. The Kentpark and Ankamall cinemas are specified with their GPS addresses.

4.3 Step 0-2 generating context observation units

We represent the sensory captured data in the form of the conceptual graph. These conceptual graphs are named the context units. Context units represent the sensory information that is related to the specific actor.

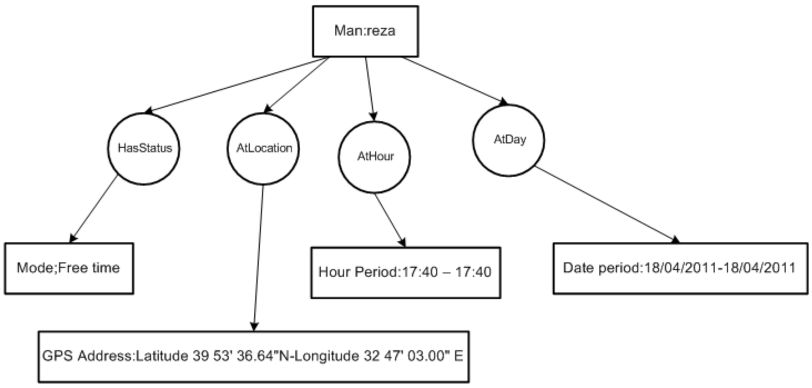

Figure 21 current context observation unit for actor reza

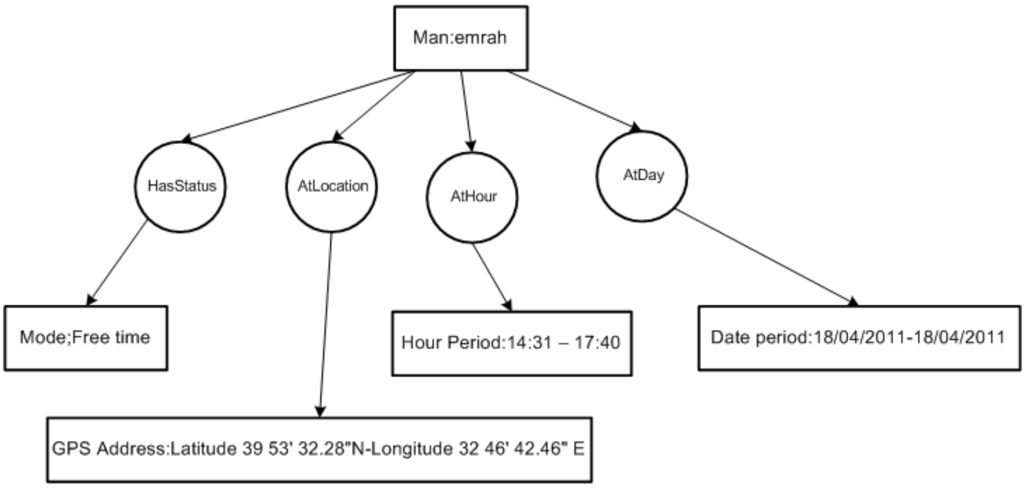

Figure 22 current context observation unit for actor Emrah

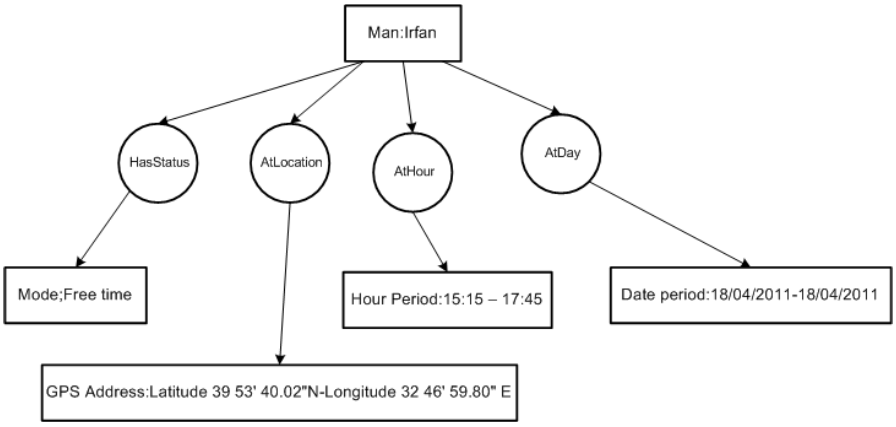

Figure 23 current context observation unit for actor Irfan

Suppose that we have the following query:

Query: If there is an interesting movie for Reza and his friends on the cinema, suggest the movie to Reza. The time of this suggestion should be after work and the status of all people should be free time.

The incoming query is given in Figure 25 as a conceptual graph. It illustrates that Reza has some friends and the status of Reza and his friends is set to free time and all of them like the same movie.

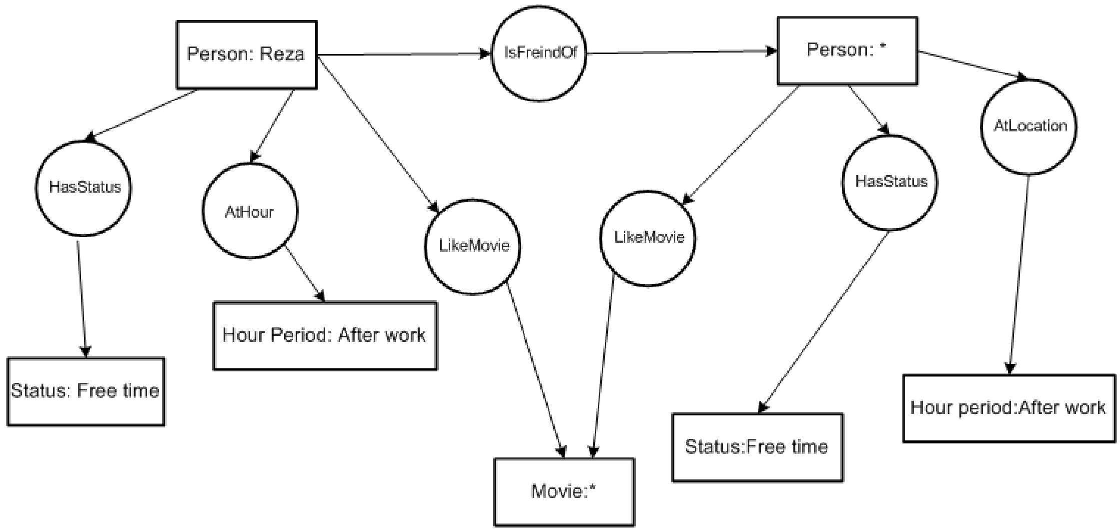

Figure 24 Incoming query in the form of query CG

4.4 Step 1) converting query CG to desired context

From the given query CG in Figure 25 the desired context CG will be generated through the following steps.

Step1-1) If there are any existential or generic referents in the query graph we translate them into individuals based on ontology. The process of the replacement is based on the small matches between the relations of desired context and the ontology. All of the existential concepts should be checked with all of the valid referents that we keep them in the concept type hierarchy. Based on the relation instances that is saved in the relation type hierarchy we have isfriend<reza,emrah> and isfriend<reza,irfan> ,so we replace the relation isfriend<person:reza,person:*> with these two relations. Updated query is represented in Figure 25.

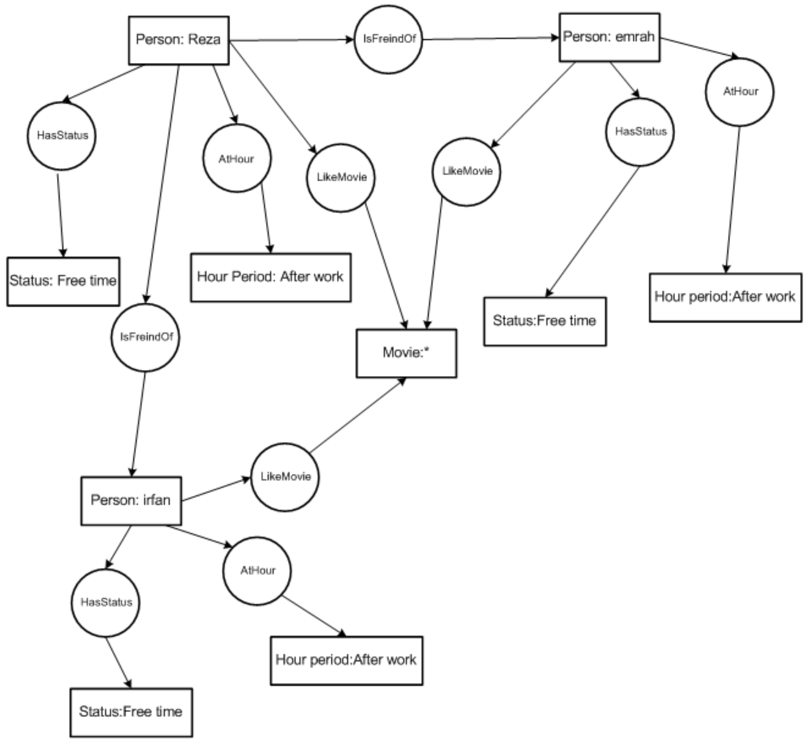

Figure 25 existential referents are replaced in the query CG

Step 1-2) the query CG should be augmented in order to contain dynamic context dimensions in addition to the static context information in order to present the perfect desired context.

The missing context dimension such as each user's location, status, hour and date is added and represented in the figure 26.

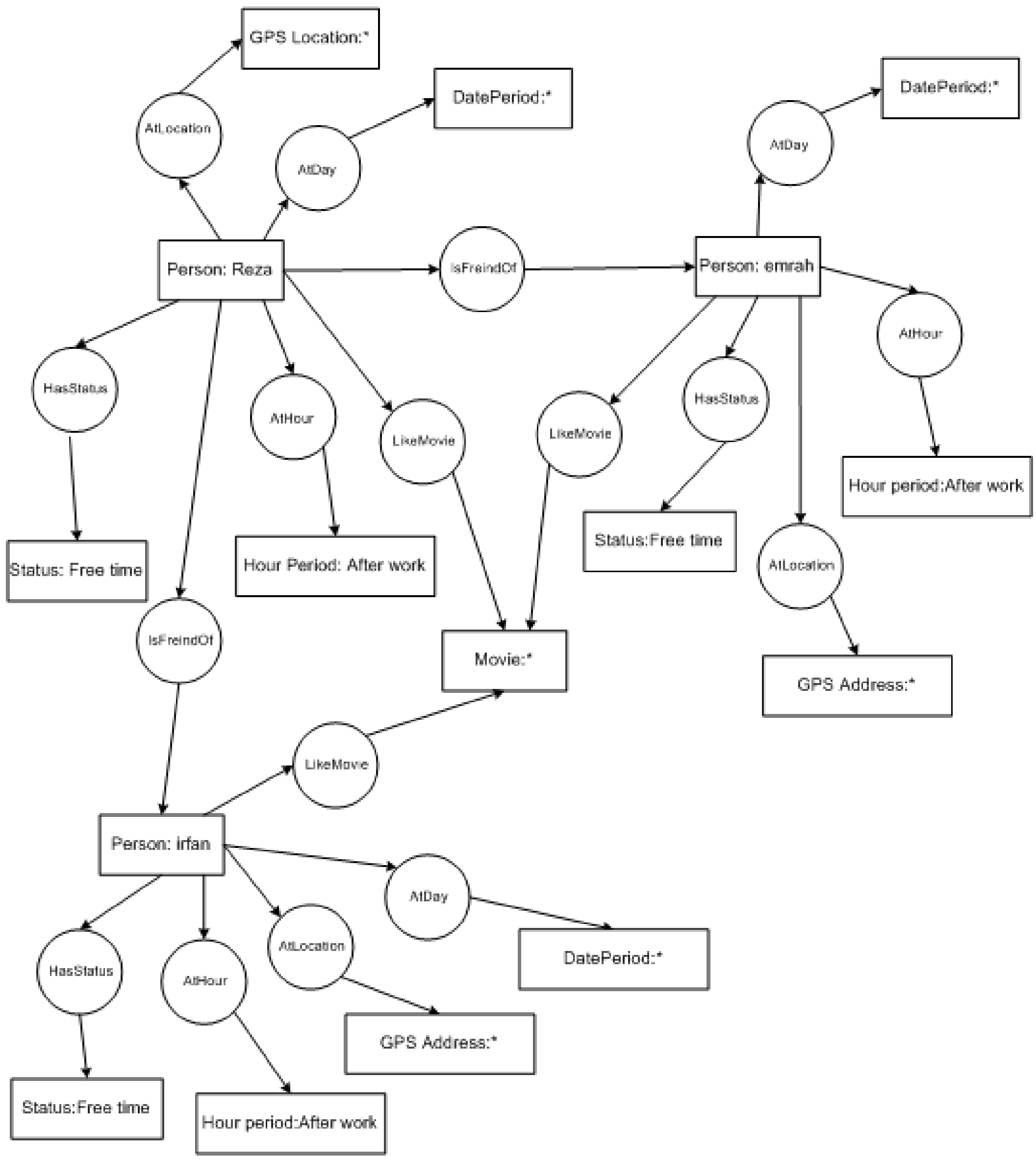

Figure 26 missing context dimensions of actors are added to the query CG

Step1-3) all of the complex relations should be replaced with more generic relations. We refer to complex relations as those relations in query CG which have no instance in the Ontology. The output of this step is given in Figure 27. (likeMovie) <person, movie> relation has no instances in the relation hierarchy, so we extend this relation in order to answer who likes which movie that is not represent in ontology in an explicit way. We extend these two relations as:

We have the (likeMovie) relation which is more specific than the LikeMovieCategory relation. Based on the relation hierarchy we have likeMovie ≤ LikeMovieCategory and based on the concept type hierarchy we have Movie≤ MovieCategory.

likeMovieCategory<person,MovieCategory>
likemovie<person,Movie>

We extend likeMovieCategory<person,MovieCategory> and likemovie<person,Movie> to :
likeMovieCategory*<person,MovieCategory,Movie>
likemovie*<person,MovieCategory,Movie>
The likeMovie* is stored in the relation type hierarchy as a new relation which is more generic than likeMovie and more specific than likeMovieCategory as :
Likemovie ≤ Likemovie* ≤ likeMovieCategory* ≤ likeMovieCategory

After extending the LikeMovie relation three new relations are generated as following:

likemovie*(person:reza,moviecategory:*,movie:*)
likemovie*(person:emrah,moviecategory:*,movie:*)
likemovie*(person:irfan,moviecategory:*,movie:*)

We set a rule here to generate the new added relations instances. After adding new relations into the relation type hierarchy we should add the relation instances into the relation hierarchy. Based on this rule, in the new added likeMoviecategory* relation the new added argument [movie:*] should be replaced according to the concept type hierarchy with respect to the referents that are sub-type of the [MovieCategory: action], so we have the following relation instances:

likeMoviecategory*(person:reza, moviecategory:action,movie:elephant white)
likeMoviecategory*(person:reza, moviecategory:action,movie:source code)
likeMoviecategory*(person:irfan, moviecategory:action,movie:elephant withe)
likeMoviecategory(person:irfan, moviecategory:action,movie:source code)

In the same way LikeMovie* relation's instances should add to the hierarchy as:
likeMovie*(person:reza, moviecategory:action, movie:elephant white)
likeMovie*(person:reza, moviecategory:action, movie:source code)
likeMovie*(person:irfan, moviecategory:action,movie:elephant withe)

likeMovie*(person:irfan, moviecategory:action,movie:source code)

Each combination adds a new relation into the query CG and is represented in Figure 27.

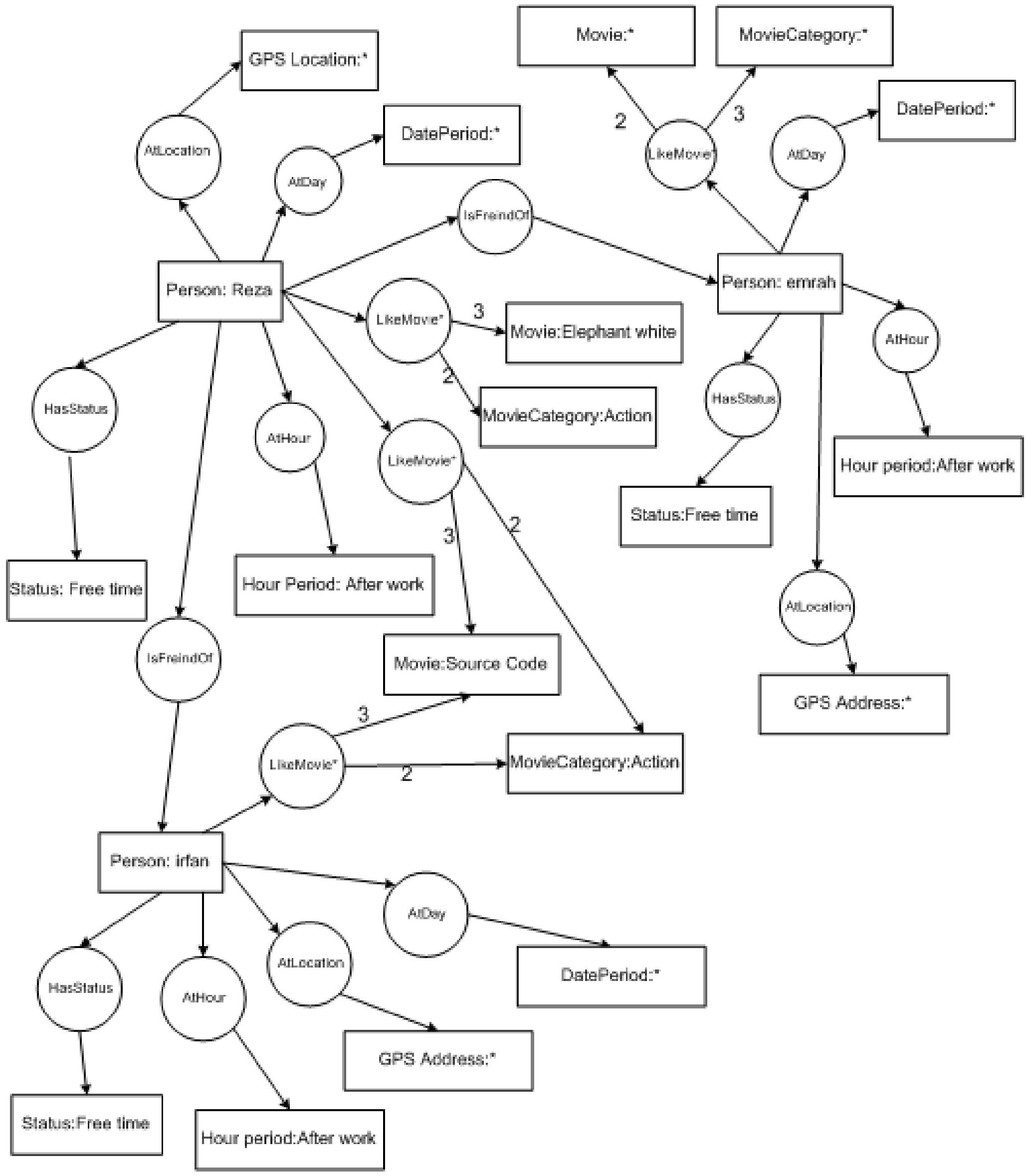

Figure 27 Complex relations are replaced in the query CG

4.5 Step 2) Filtering the current context observations

Too many context observations may exist for each actor, in order to benefit from them we need to filter them based on the requested context in the desired context. The filtering process should be done by comparing each context observation with the desired context. The matching may have more than one acceptable result, especially when the desired context defines broad ranges for date and hour. We identify all of the results as the valid context observations. These context observations will be used to generate the current context CG in the next step.

For filtering the history of current context observations in regard to a passive query all of current context observations compare with the desired context CG.

The current context observation units for actors of the desired context CG (Figure 27) Reza, Emrah and Irfan are given in Figures 21, 22, 23.

4.6 Step 3) generating current context CG

The main idea for generating the current context CG is to include those contexts that have the most similarity to the desired context in the form of conceptual graph. The context observations that are selected in the previous step will be used to generate the current context CG. We create a copy of the desired context CG and update the context dimensions of each actor in order to form the current context CG. The resultant current context CG is given in Figure 28.

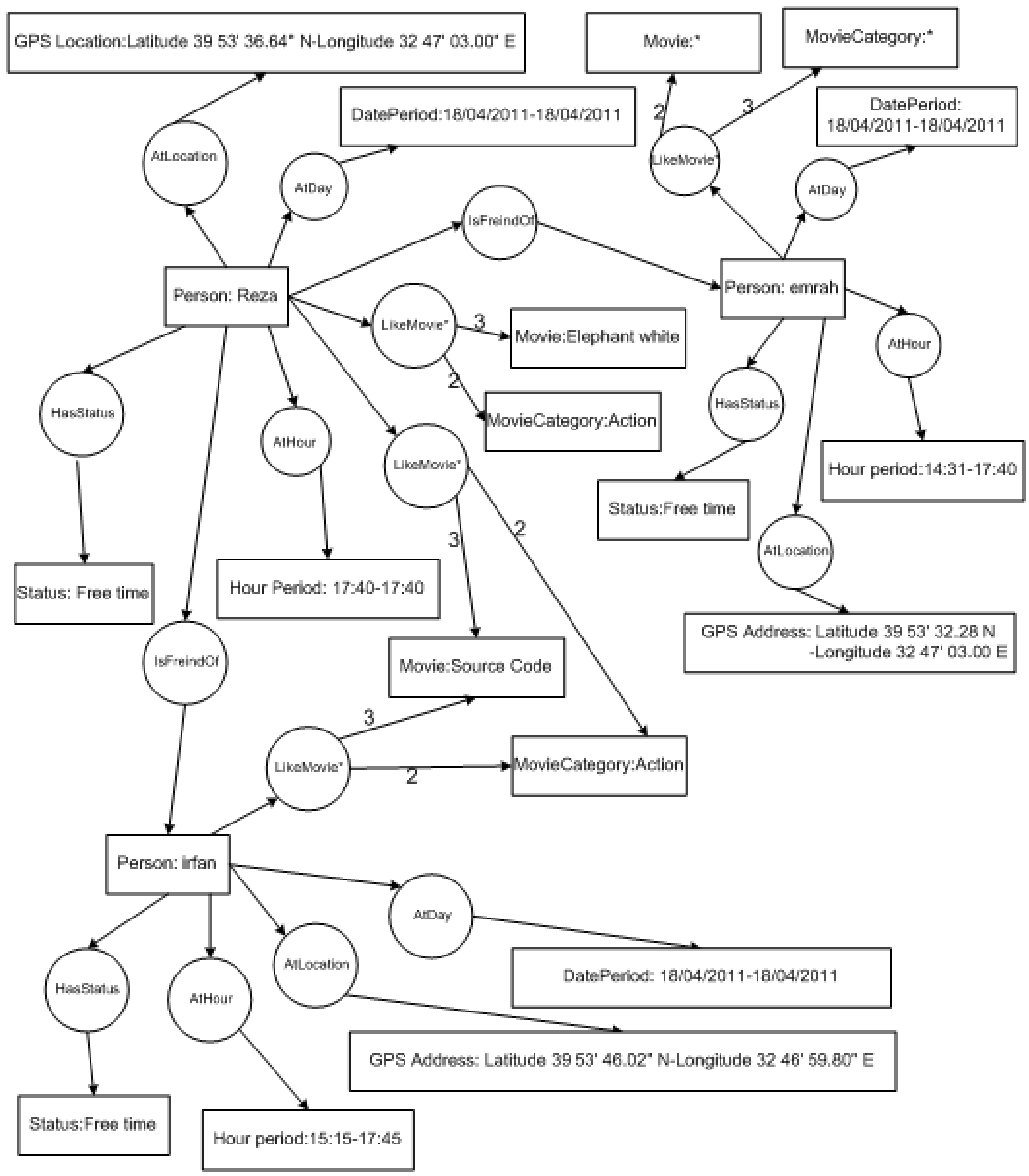

Figure 28 current context CG that is generated based on desired context CG.

4.7 Step 4) matching between the current context and desired context

As the last step we match the generated current context CG and the desired CG together.

We use the following abbreviations during the computing similarity

R= Latitude 39 53’ 36.64” N-Longitude 32 47’ 03.00” E

E= Latitude 39 53’ 32.28” N-Longitude 32 46’ 42.46” E

I= Latitude 39 53’ 46.02” N-Longitude 32 46’ 59.80” E

Details of the matching are as follows:

- r_{d1}= atloccation1<person: Reza, GPS Address: *>
- r_{c1}= atlocation1<person: Reza, GPS Address: R >
- $Sim_{relation\ type}$(atlocation,atlocation)=1
- $Sim_{concept\ type}$ (person,person)=1
- $Sim_{Referent}$ (reza, reza)=1
- $Sim_{concept\ type}$ (GPS Address, GPS Address)=1
- $Sim_{Referent}$ (*, R)=1

$Sim_{relation\ type}$(atLocation, atLocation)* ((Sim $_{concept\ type}$ (person,person)*$Sim_{referent}$(reza,reza) + Sim $_{concept\ type}$ (GPS address, GPS address)*Sim $_{referent}$(*, R))/2=1

  - r_{d2}=athour < person:reza, hour period:after work> // current hour:17:40
  - r_{c2}=athour < person:reza ,hour period:17:40 pm-17:40 pm >
  - $Sim_{relation\ type}$(athour,athour)=1
  - $Sim_{concept\ type}$ (person, person)=1
  - $Sim_{referent}$ (reza,reza)=1
  - $Sim_{concept\ type}$ (hour period, hour period)=1
  - $Sim_{referent}$ (after work,17:40 pm-17:40 pm)=1

  $Sim_{relation\ type}$(athour, athour)* ((Sim $_{concept\ type}$(person,person)*Sim-$_{referent}$(reza,reza) + Sim $_{concept\ type}$ (hour period, hour period)*Sim $_{referent}$(after work, 17:40 pm-17:40 pm))/2=1

    - r_{d3}=atday<person:reza, date period:* > //current date18/04/2011
    - r_{c3}=atday<person:reza, date period:18/04/2011-18/04/2011>
    - $Sim_{relation\ type}$(Atday, Atday)=1
    - Sim $_{concept\ type}$(person,person)=1
    - $Sim_{referent}$(reza,reza)=1
    - Sim $_{concept\ type}$(date period, date period)=1
    - $Sim_{referent}$ (*, 18/04/2011-18/04/2011)=1

Sim$_{\text{relation type}}$(atday, atday)* ((Sim $_{\text{concept type}}$(person,person)*Sim-$_{\text{referent}}$(reza,reza) + Sim $_{\text{concept type}}$ (date period, date period)*Sim $_{\text{referent}}$(*, 18/04/2011-18/04/2011))/2=1

- r_{d4}=hasstatus<person:reza, status:free time>
- r_{c4}=hasstatus<person:reza, mood:free time>
- Sim$_{\text{relation type}}$(hasStatus, hasStatus)=1
- Sim $_{\text{concept type}}$(person,person)=1
- Sim$_{\text{referent}}$(reza,reza)=1
- Sim $_{\text{concept type}}$(Status, mood)=1
- Sim $_{\text{referent}}$ (free time,free time)=1

Sim$_{\text{relation type}}$(hasStatus, hasStatus)* ((Sim $_{\text{concept type}}$(person,person)*Sim-$_{\text{referent}}$(reza,reza) + Sim $_{\text{concept type}}$ (Status, mood)*Sim $_{\text{referent}}$(free time, free time))/2=1

- r_{d5}=isfriendof<person:reza, person:emrah>
- r_{c5}= isfriendof<person:reza, person:emrah>
- Sim$_{\text{relation type}}$(isfriendof, isfriendof)=1
- Sim $_{\text{concept type}}$(person,person)=1
- Sim$_{\text{referent}}$(reza,reza)=1
- Sim $_{\text{concept type}}$(person, person)=1
- Sim $_{\text{referent}}$ (emrah,emrah)=1

Sim$_{\text{relation type}}$(isfriendof, isfriendof)* ((Sim $_{\text{concept type}}$(person,person)*Sim-$_{\text{referent}}$(reza,reza) + Sim $_{\text{concept type}}$ (person, person)*Sim $_{\text{referent}}$(emrah, emrah))/2=1

- r_{d6}=isfriendof<person:reza, person:irfan>
- r_{c6}= isfriendof<person:reza, person:irfan>
- Sim$_{\text{relation type}}$(isfriendof, isfriendof)=1
- Sim $_{\text{concept type}}$(person,person)=1
- Sim$_{\text{referent}}$(reza,reza)=1
- Sim $_{\text{concept type}}$(person, person)=1
- Sim $_{\text{referent}}$ (irfan,irfan)=1

Sim$_{\text{relation type}}$(isfriendof, isfriendof)* ((Sim $_{\text{concept type}}$(person,person)*Sim-$_{\text{referent}}$(reza,reza) + Sim $_{\text{concept type}}$ (person, person)*Sim $_{\text{referent}}$(irfan,irfan))/2=1

- r_{d7}=likeMovie*<person:reza,MovieCategory:Action,movie:"source code">

- r_{c7}= likeMovie*<person:reza,MovieCategory:Action,movie:”source code”>
- $Sim_{relation\ type}$(likeMovie*, likeMovie*)=1
- Sim $_{concept\ type}$(person,person)=1
- $Sim_{referent}$(reza,reza)=1
- Sim $_{concept\ type}$(MovieCategory, MovieCategory)=1
- $Sim_{referent}$(action,action)=1
- Sim $_{concept\ type}$(movie,movie)=1
- $Sim_{referent}$(source code,source code)=1

$Sim_{relation\ type}$(LikeMovie*, LikeMovie*)* ((Sim $_{concept\ type}$(person,person)*$Sim_{referent}$(reza,reza) + Sim $_{concept\ type}$ (MovieCategory, MovieCategory)*Sim $_{referent}$(action,action) + Sim $_{concept\ type}$ (Movie, Movie)*Sim $_{referent}$(source code, source code))/3=1

- r_{d8}=likeMovie*<person:reza,MovieCategory:Action,movie:”elephant white”>
- r_{c8}= likeMovie*<person:reza,MovieCategory:Action,movie:”elephant white”>
- $Sim_{relation\ type}$(likeMovie*, likeMovie*)=1
- Sim $_{concept\ type}$(person,person)=1
- $Sim_{referent}$(reza,reza)=1
- Sim $_{concept\ type}$(MovieCategory, MovieCategory)=1
- $Sim_{referent}$(action,action)=1
- Sim $_{concept\ type}$(movie,movie)=1
- $Sim_{referent}$(elephant white, elephant white)=1

$Sim_{relation\ type}$(LikeMovie*, LikeMovie*)* ((Sim $_{concept\ type}$(person,person)*$Sim_{referent}$(reza,reza) + Sim $_{concept\ type}$ (MovieCategory, MovieCategory)*Sim $_{referent}$(action,action) + Sim $_{concept\ type}$ (Movie, Movie)*Sim $_{referent}$(elephant white, elephant white))/3=1

- r_{d9}=likeMovie*<person:irfan,MovieCategory:Action,movie:”source code”>
- r_{c9}= likeMovie*<person:irfan,MovieCategory:Action,movie:”source code”>
- $Sim_{relation\ type}$(likeMovie*, likeMovie*)=1
- Sim $_{concept\ type}$(person,person)=1
- $Sim_{referent}$(irfan,irfan)=1
- Sim $_{concept\ type}$(MovieCategory, MovieCategory)=1
- $Sim_{referent}$(action,action)=1

- Sim $_{concept\ type}$(movie,movie)=1
- Sim$_{referent}$(source code,source code)=1

Sim$_{relation\ type}$(LikeMovie*, LikeMovie*)* ((Sim $_{concept\ type}$(person,person)*Sim$_{referent}$(irfan,irfan) + Sim $_{concept\ type}$ (MovieCategory, MovieCategory)*Sim $_{referent}$(action,action) + Sim $_{concept\ type}$ (Movie, Movie)*Sim $_{referent}$(source code, source code))/3=1

- r$_{d10}$= atloccation<person: irfan, GPS Address: *>
- r$_{c10}$= atlocation<person: irfan, GPS Address: I >
- Sim$_{relation\ type}$(atlocation,atlocation)=1
- Sim $_{concept\ type}$ (person,person)=1
- Sim $_{Referent}$ (irfan,irfan)=1
- Sim $_{concept\ type}$ (GPS Address, GPS Address)=1
- Sim $_{Referent}$ (*, I)=1

Sim$_{relation\ type}$(atLocation, atLocation)* ((Sim $_{concept\ type}$ (person,person)*Sim$_{referent}$(irfan,irfan) + Sim $_{concept\ type}$ (GPS address, GPS address)*Sim $_{referent}$(*, I))/2=1

  - r$_{d11}$=athour < person:irfan, hour period:after work> // current hour:17:40
  - r$_{c11}$=athour < person:irfan ,hour period:15:15 pm-17:40 pm >
  - Sim$_{relation\ type}$(athour,athour)=1
  - Sim $_{concept\ type}$ (person, person)=1
  - Sim $_{referent}$ (irfan,irfan)=1
  - Sim $_{concept\ type}$ (hour period, hour period)=1
  - Sim $_{referent}$ (after work,15:15 pm-17:40 pm)=1

  Sim$_{relation\ type}$(athour, athour)* ((Sim $_{concept\ type}$(person,person)*Sim-$_{referent}$(irfan,irfan) + Sim $_{concept\ type}$ (hour period, hour period)*Sim $_{referent}$(after work, 15:15 pm-17:40 pm))/2=1

    - r$_{d12}$=atday<person:irfan, date period:* > //current date18/04/2011
    - r$_{c12}$=atday<person:irfan, date period:18/04/2011-18/04/2011>
    - Sim$_{relation\ type}$(Atday, Atday)=1
    - Sim $_{concept\ type}$(person,person)=1
    - Sim$_{referent}$(irfan,irfan)=1
    - Sim $_{concept\ type}$(date period, date period)=1

- $Sim_{referent}$ (*, 18/04/2011-18/04/2011)=1

  $Sim_{relation\ type}$(atday, atday)* (($Sim_{concept\ type}$(person,person)*$Sim_{referent}$(irfan,irfan)+ $Sim_{concept\ type}$ (date period, date period)*$Sim_{referent}$(*, 18/04/2011-18/04/2011))/2=1

- r_{d13}=hasstatus<person:reza, status:free time>
- r_{c13}=hasstatus<person:reza, mood:free time>
- $Sim_{relation\ type}$(hasStatus, hasStatus)=1
- $Sim_{concept\ type}$(person,person)=1
- $Sim_{referent}$(irfan,irfan)=1
- $Sim_{concept\ type}$(Status, mood)=1
- $Sim_{referent}$ (free time,free time)=1

$Sim_{relation\ type}$(hasStatus, hasStatus)* (($Sim_{concept\ type}$(person,person)*$Sim_{referent}$(irfan,irfan) + $Sim_{concept\ type}$ (Status, mood)*$Sim_{referent}$(free time, free time))/2=1

- r_{d14}=likeMovie*<person:irfan,MovieCategory:Action,movie:"elephant white">
- r_{c14}= likeMovie*<person:irfan,MovieCategory:Action,movie:"elephant white">
- $Sim_{relation\ type}$(likeMovie*, likeMovie*)=1
- $Sim_{concept\ type}$(person,person)=1
- $Sim_{referent}$(irfan,irfan)=1
- $Sim_{concept\ type}$(MovieCategory, MovieCategory)=1
- $Sim_{referent}$(action,action)=1
- $Sim_{concept\ type}$(movie,movie)=1
- $Sim_{referent}$(elephant white, elephant white)=1

  $Sim_{relation\ type}$(LikeMovie*, LikeMovie*)* (($Sim_{concept\ type}$(person,person)*$Sim_{referent}$(reza,reza) + $Sim_{concept\ type}$ (MovieCategory, MovieCategory)*$Sim_{referent}$(action,action) + $Sim_{concept\ type}$ (Movie, Movie)*$Sim_{referent}$(elephant white, elephant white))/3=1

- r_{d15}= atloccation<person:emrah, GPS Address: *>
- r_{c15}= atlocation<person: Emrah, GPS Address: E >
- $Sim_{relation\ type}$(atlocation,atlocation)=1
- $Sim_{concept\ type}$ (person,person)=1

- $Sim_{Referent}$ (emrah,emrah)=1
- $Sim_{concept\ type}$ (GPS Address, GPS Address)=1
- $Sim_{Referent}$ (*, E)=1

$\underline{Sim_{relation\ type}(atLocation, atLocation)* ((Sim_{concept\ type} (person,person)*Sim_{referent}(emrah,emrah) + Sim_{concept\ type} (GPS\ address, GPS\ address)*Sim_{referent}(*, E))/2=1}$

  - r_{d16}=athour < person:emrah, hour period:after work> // current hour:17:40
  - r_{c16}=athour < person:emrah ,hour period:14:31 pm-17:40 pm >
  - $Sim_{relation\ type}$(athour,athour)=1
  - $Sim_{concept\ type}$ (person, person)=1
  - $Sim_{referent}$ (emrah,emrah)=1
  - $Sim_{concept\ type}$ (hour period, hour period)=1
  - $Sim_{referent}$ (after work,14:31 pm-17:40 pm)=1

  $\underline{Sim_{relation\ type}(athour, athour)* ((Sim_{concept\ type}(person,person)*Sim_{referent}(emrah,emrah) + Sim_{concept\ type} (hour\ period, hour\ period)*Sim_{referent}(after\ work, 14:31\ pm-17:40\ pm))/2=1}$

    - r_{d17}=atday<person:emrah, date period:* > //current date18/04/2011
    - r_{c17}=atday<person:emrah, date period:18/04/2011-18/04/2011>
    - $Sim_{relation\ type}$(Atday, Atday)=1
    - $Sim_{concept\ type}$(person,person)=1
    - $Sim_{referent}$(emrah,emrah)=1
    - $Sim_{concept\ type}$(date period, date period)=1
    - $Sim_{referent}$ (*, 18/04/2011-18/04/2011)=1

    $\underline{Sim_{relation\ type}(atday, atday)* ((Sim_{concept\ type}(person,person)*Sim_{referent}(irfan,irfan)+ Sim_{concept\ type} (date\ period, date\ period)*Sim_{referent}(*, 18/04/2011-18/04/2011))/2=1}$

    - r_{d18}=hasstatus<person:emrah, status:free time>
    - r_{c18}=hasstatus<person:emrah, mood:free time>
    - $Sim_{relation\ type}$(hasStatus, hasStatus)=1
    - $Sim_{concept\ type}$(person,person)=1
    - $Sim_{referent}$(emrah,emrah)=1
    - $Sim_{concept\ type}$(Status, mood)=1
    - $Sim_{referent}$ (free time,free time)=1

Sim$_{relation\ type}$(hasStatus, hasStatus)* ((Sim $_{concept\ type}$(person,person)*Sim-$_{referent}$(emrah,emrah) + Sim $_{concept\ type}$ (Status, mood)*Sim $_{referent}$(free time, free time))/2=1

- r_{d19}=likeMovie*<person:emrah,MovieCategory:*,movie:*>
- r_{c19}= likeMovie*<person:emrah,MovieCategory:*,movie:*>
- Sim$_{relation\ type}$(likeMovie*, likeMovie*)=1
- Sim $_{concept\ type}$(person,person)=1
- Sim$_{referent}$(emrah,emrah)=1
- Sim $_{concept\ type}$(MovieCategory, MovieCategory)=1
- Sim$_{referent}$(action,action)=1
- Sim $_{concept\ type}$(movie,movie)=1
- Sim$_{referent}$(*, *)=1

Sim$_{relation\ type}$(LikeMovie*, LikeMovie*)* ((Sim $_{concept\ type}$(person,person)*Sim-$_{referent}$(emrah,emrah) + Sim $_{concept\ type}$ (MovieCategory, MovieCategory)*Sim $_{referent}$(*,*) + Sim $_{concept\ type}$ (Movie, Movie)*Sim $_{referent}$(*,*))/3=1

Sim$_{CG}$(D,C)=AVG(Sim$_{relation}$(r_{d1},r_{c1}), Sim$_{relation}$(r_{d2},r_{c2}), Sim$_{relation}$(r_{d3},r_{c3}), Sim$_{relation}$(r_{d4},r_{c4}), Sim$_{relation}$(r_{d5},r_{c5}), Sim$_{relation}$(r_{d6},r_{c6}), Sim$_{relation}$(r_{d7},r_{c7}), Sim$_{relation}$(r_{d8},r_{c8}), Sim$_{relation}$(r_{d9},r_{c9}), Sim$_{relation}$(r_{d10},r_{c10}), Sim$_{relation}$(r_{d11},r_{c11}), Sim$_{relation}$(r_{d12},r_{c12}), Sim$_{relation}$(r_{d13},r_{c13}), Sim$_{relation}$(r_{d14},r_{c514}), Sim$_{relation}$(r_{d15},r_{c15}), Sim$_{relation}$(r_{d16},r_{c16}), Sim$_{relation}$(r_{d17},r_{c17}), Sim$_{relation}$(r_{d18},r_{c18}), Sim$_{relation}$(r_{d19},r_{c19}))=1

This result shows that the current context observations satisfy the criteria of the desired context, so we can suggest the current context to answer the query as:

You have Free time at 17:40- 18/04/2011 Movie: Action,” elephant white” Movie: Action, ”source code” Friend: Emrah, free time, 17:40-18/04/2011

CHAPTER 5

CONCLUSION

5.1 Summary

In this thesis, we study the context-aware applications area with special concerns about context representation methods and utilization of Ontology in context matching. Definition of context and the different dimension of context are figured out. Different aspects of context-aware applications and the role of context matching are mentioned. The related works in context matching are examined. After giving the background information about the conceptual graphs we present our approach in order to utilize the conceptual graph in context representation. We also define the benefits of Ontology and present how to use conceptual graphs to represent ontology.

The main problem in the context matching is the lack of a best matching approach that supports different context dimensions. In regard to multidimensionality of context information we propose to use ontology in order to represent long term contextual information of desired context and current context.

We propose an algorithm to match the desired context and current context which are represented in conceptual graph format. Our matching algorithm is defined to match two conceptual graphs. We represent the current context and desired context both in the conceptual graph format.

Some steps are defined prior to matching in order to create the desired context conceptual graph based on the incoming query and ontological information. The current context conceptual graph is generated based on the desired context CG structure and the context observation units.

Some steps are introduced to replace the generic and existential concept with the appropriate ones from the ontology and an approach is proposed to extend complex relations in the incoming query in order to replace them with generic relations with instances in the ontology.

In order to generate the current context conceptual graph the generated structure of the desired context CG should be updated with selected current context observation units which are filtered based on the actors in the query.

The similarity score between desired context and the current context is defined as the average of relations similarity. A weighted similarity score is also proposed in order to distinguish more important relations from the less important ones.

The similarity between two relations depends on the type similarity of relations and the conceptual similarity of relations. Concepts in a relation are defined as composition of concept types and referents. We take concept type similarity and referent similarity into account in order to define the conceptual similarity.

The relations are compared in pairs and the average of the similarities of those relations with the highest similarity is defined as the similarity score. The similarity score is between 0 and 1. The similarity score 1 indicates the highest similarity and 0 indicate no similarity.

5.2 Discussion

The algorithm for best matching contextual information presented in this thesis has three important features:

1) It uses the conceptual graphs as the formal method for representing context.
2) It uses extended ontology as the knowledge base of the domain and represents it with conceptual graphs, the significance of this method of representation embedded in functionality and reasoning ability of the ontology in contrast with the static ontology representation. In

addition the relation extension and ability to complete missing arguments in a relation enhanced formalism for translating complex relations. In extended ontology the concept type hierarchy is extended to (T_C U I_C) and the relation type hierarchy is extended to (T_R U I_R).This enables us to answer those queries with complex relations that have no explicit instances in the ontology. We extend the concept type hierarchy like the relation type hierarchy in order to map sensory data into real world concepts.

3) It uses best matching approach. The matching process is based on the CG matching rules. The semantic similarity between different concept types and relation types are defined according to the previous works in the conceptual graph area. The result of matching is represented as a granular similarity score which is represented in a range between 0 to 1.

4) It formalizes some steps prior to matching process in order to generate appropriate desired context and current context. The concepts in the incoming query with universal or existential referents are replaced with their equivalences from the ontology. The complex relations or those relations with no instances in the ontology are replaced with a generic relation that is generated based on an extension on complex relation. Relation extension is demonstrated as Ontology reasoning tool to convert the complex relations.

5) Semantic similarity between different referents is defined based on their positions in the concept type hierarchy. For those non-numeric concepts with isolated positions in the concept type hierarchy the relational similarity is defined. For numeric concepts depending on the domain of the problem their distance from each other is mapped to a similarity score.

6) The proposed algorithm can also be used for historical context searching. The current context that consists of context observation can be filtered based on desired context. If in the desired context some historical context is mentioned then the current context observations should be selected based on these observations.

7) The proposed context matching approach is an integration of prior works in semantic search by matching conceptual graphs, ontology extension based on conceptual graphs and weighted granular best matching algorithm for context-aware applications. We use the main idea of using conceptual graph as the best tool for representing context. The extended ontology is used to clarify those complex relations that may occur in the incoming query. An approach to score the similarity between two contexts is designed in order to satisfy the best matching principles.

8) Our work is theoretical. We have not implemented the proposed algorithm.

5.3 Future work

For context information that comes from different sensors the reliability of the sensors should be considered and also an approach for aggregating the various context observations as a unique one should be proposed.

Implementation of this idea and building software tools for a real-world application and addressing the HCI issues are the possible future works. We are planning to use PROLOG+CG which is a powerful development environment that facilitate conceptual graph processing.(Kabbaj, Moulin, Gancef, Nadeau, & Rouleau, 2001)

A conjunction method for information fusion as the complementary step should be utilized. The partial information represented in the form of desired context and the matched current context CG should be represented in a unified format. The matching process can be used for checking their consistency.

The conceptual graphs considered in this thesis can be augmented with first order logic to improve logical reasoning for matched context.

REFERENCES

Abowd, G. D., Atkeson, C. G., Hong, J., Long, S., Kooper, R., & Pinkerton, M. (1997). Cyberguide: A mobile context-aware tour guide. Wireless Networks, 3(5), 421–433.

Amati, G., & Ounis, I. (2000). Conceptual graphs and first order logic. The Computer Journal, 43(1), 1.

Baldauf, M., Dustdar, S., & Rosenberg, F. (2007). A survey on context-aware systems. International Journal of Ad Hoc and Ubiquitous Computing, 2(4), 263–277.

Bederson, B. B. (1995). Audio augmented reality: a prototype automated tour guide. Conference companion on Human factors in computing systems (p. 210–211).

Beigl, M. (2000). MemoClip: A location-based remembrance appliance. Personal and Ubiquitous Computing, 4(4), 230–233.

Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A., & Riboni, D. (2010). A survey of context modelling and reasoning techniques. Pervasive and Mobile Computing, 6(2), 161–180.

Brézillon, P. (1999). Context in Artificial Intelligence: II. Key elements of contexts. Computers and Artificial Intelligence, 18, 425–446.

Brotherton, J. A., Abowd, G. D., & Truong, K. N. (1998). Supporting capture and access interfaces for informal and opportunistic meetings. Georgia Institute of Technology Technical Report: GIT-GVU-99, 6.

Brown, P. J., Bovey, J. D., & Xian Chen. (1997). Context-aware applications: from the laboratory to the marketplace. Personal Communications, IEEE, 4(5), 58-64. doi:10.1109/98.626984

Chein, M., & Mugnier, M. L. (2008). Graph-based knowledge representation: computational foundations of conceptual graphs. Springer-Verlag New York Inc.

Chen, G. and Kotz, D. (2000). A survey of context-aware mobile computing research. Dartmouth College, Computer Science. Retrieved from http://www.cs.dartmouth.edu/reports/TR2000-381.ps.Z

Chen, H., Finin, T., & Joshi, A. (2003). An ontology for context-aware pervasive computing environments. The Knowledge Engineering Review, 18(03), 197–207.

Corbett, D. (2003). Reasoning and unification over conceptual graphs. Springer Us.

Corbett, D. (2004). Interoperability of ontologies using conceptual graph theory. Conceptual Structures at Work, 236–236.

Corbett, D. (2008). Graph-Based Representation and Reasoning for Ontologies. Computational Intelligence: A Compendium, 351–379.

Corbett, D., & Mayer, W. (2005). Concept type hierarchy as ontology: an example historical knowledge base. AI 2004: Advances in Artificial Intelligence, 251–266.

Davies, N., Mitchell, K., Cheverst, K., & Blair, G. (1998). Developing a context sensitive tourist guide. First workshop on human computer interaction with mobile devices, GIST Technical Report G (Vol. 98).

Dey, A. K. (1998). Context-aware computing: The CyberDesk project. Proceedings of the AAAI 1998 Spring Symposium on Intelligent Environments (p. 51–54).

Dey, A. K. (2000). Providing architectural support for building context-aware applications. Citeseer.

Dey, A. K., & Abowd, G. D. (2000). Towards a better understanding of context and context-awareness. CHI 2000 workshop on the what, who, where, when, and how of context-awareness (Vol. 4, p. 1–6).

Dey, A. K., Abowd, G. D., & Salber, D. (1999). A context-based infrastructure for smart environments. Proceedings of the 1st International Workshop on Managing Interactions in Smart Environments (MANSE'99) (p. 114–128).

Dey, A. K., Abowd, G. D., & Salber, D. (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. Human–Computer Interaction, 16(2), 97–166.

Dourish, P., Edwards, W. K., LaMarca, A., Lamping, J., Petersen, K., Salisbury, M., Terry, D. B., et al. (2000). Extending document management systems with user-specific active properties. ACM Transactions on Information Systems (TOIS), 18(2), 140–170.

Feiner, S., MacIntyre, B., H\öllerer, T., & Webster, A. (1997). A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment. Personal and Ubiquitous Computing, 1(4), 208–217.

Fels, S., Sumi, Y., Etani, T., Simonet, N., Kobayshi, K., & Mase, K. (1998). Progress of C-MAP: A context-aware mobile assistant. Proceeding of AAAI 1998 Spring Symposium on Intelligent Environments, Technical Report SS-98-02.

Foo, N., Garner, B. J., Rao, A., & Tsui, E. (1992). Semantic distance in conceptual graphs. Conceptual Structures: current research and practice, 149–154.

Genesereth, M. R. (1998). Knowledge interchange format. draft proposed american national standard(dpans). ncits. t2/98-004.

Gu, T., Pung, H. K., & Zhang, D. Q. (2005a). A service-oriented middleware for building context-aware services. Journal of Network and Computer Applications, 28(1), 1–18.

Gu, T., Pung, H. K., & Zhang, D. Q. (2005b). A service-oriented middleware for building context-aware services. Journal of Network and Computer Applications, 28(1), 1–18.

Harrison, B. L., Fishkin, K. P., Gujar, A., Mochon, C., & Want, R. (1998). Squeeze me, hold me, tilt me! An exploration of manipulative user interfaces. Proceedings of the SIGCHI conference on Human factors in computing systems (p. 17–24).

Healey, J., & Picard, R. W. (2002). Startlecam: A cybernetic wearable camera. Wearable Computers, 1998. Digest of Papers. Second International Symposium on (p. 42–49).

Heflin, J. (2004). OWL Web Ontology Language-Use Cases and Requirements. W3C Recommendation, 10.

Heiner, J. M., Hudson, S. E., & Tanaka, K. (1999). The information percolator: ambient information display in a decorative object. Proceedings of the 12th annual ACM symposium on User interface software and technology (p. 141–148).

Hertz. (n.d.). Hertz Neverlost Review. Retrieved January 22, 2011, from http://www.bythom.com/neverlost.htm

Hinckley, K., Pierce, J., Sinclair, M., & Horvitz, E. (2000). Sensing techniques for mobile interaction. Proceedings of the 13th annual ACM symposium on User interface software and technology (p. 91–100).

Hong, J.-yi, Suh, E.-ho, & Kim, S.-J. (2009). Context-aware systems: A literature review and classification. Expert Systems with Applications, 36(4), 8509 - 8522. doi:DOI: 10.1016/j.eswa.2008.10.071

Humanistic Informatics at Department of Communication, Aalborg University, Denmark. (n.d.). Onlin Course in Knowledge Representation using Conceptual Graphs. Onlinet http://cg.huminf.aau.dk/. Retrieved from http://cg.huminf.aau.dk/

Ishii, H., & Ullmer, B. (1997). Tangible bits: towards seamless interfaces between people, bits and atoms. Proceedings of the SIGCHI conference on Human factors in computing systems (p. 234–241).

Jena Semantic Web Framework. (n.d.). Online http://jena.sourceforge.net/. Retrieved from http://jena.sourceforge.net/

Kabbaj, A., Moulin, B., Gancef, J., Nadeau, D., & Rouleau, O. (2001). Uses, improvements, and extensions of Prolog+ CG: Case studies. Conceptual Structures: Broadening the Base, 346–359.

KOCABALLI, A. B. A. K. (2005). weighted granular best matching algorithm for context-aware computing systems (M.Sc. thesis). MIDDLE EAST TECHNICAL UNIVERSITY.

Korpipaa, P., & Mantyjarvi, J. (2003). An ontology for mobile device sensor-based context awareness. Modeling and Using Context, 451–458.

Korpipaa, P., Mantyjarvi, J., Kela, J., Keranen, H., & Malm, E. J. (2003). Managing context information in mobile devices. Pervasive Computing, IEEE, 2(3), 42-51. doi:10.1109/MPRV.2003.1228526

Lamming, M., & Flynn, M. (1994). Forget-me-not: Intimate computing in support of human memory. Proceedings of FRIEND21 (Vol. 94, p. 2–4).

Laudy, C., & Ganascia, J. G. (2010). Introducing semantic knowledge in high-level fusion. Military Communications Conference, 2009. MILCOM 2009. IEEE (p. 1–7).

Marmasse, N., & Schmandt, C. (2000). Location-aware information delivery with commotion. Handheld and Ubiquitous Computing (p. 361–370).

McCarthy, J., & Anagnost, T. (2000). EventManager: Support for the peripheral awareness of events. Handheld and Ubiquitous Computing (p. 67–96).

McCarthy, J., & Meidel, E. (1999). ACTIVE MAP: A visualization tool for location awareness to support informal interactions. Handheld and Ubiquitous Computing (p. 158–170).

Miraoui, M., Tadj, C., & Ben Amar, C. (2008). Architectural Survey of Context-Aware Systems in Pervasive Computing Environment. Ubiquitous Computing and Communication Journal, 3(3).

Montes-y-Gómez, M., López-López, A., & Gelbukh, A. (2000). Information retrieval with conceptual graph matching. Database and Expert Systems Applications (p. 312–321).

Mynatt, E. D., Back, M., Want, R., Baer, M., & Ellis, J. B. (1998). Designing audio aura. Proceedings of the SIGCHI conference on Human factors in computing systems (p. 566–573).

Nguyen, P., Kaneiwa, K., Corbett, D., & Nguyen, M. Q. (2008). An ontology formalization of relation type hierarchy in conceptual structure theory. AI 2008: Advances in Artificial Intelligence, 79–85.

Noy, N. F., McGuinness, D. L., & others. (2001). Ontology development 101: A guide to creating your first ontology. Citeseer.

Pascoe, J., Ryan, N. S., & Morse, D. R. (1998). Human computer giraffe interaction: HCI in the field.

Pawar, P., & Tokmakoff, A. (2006). Ontology-based context-aware service discovery for pervasive environments.

Poslad, S. (2009). Ubiquitous computing: smart devices, environments and interactions. John Wiley & Sons Inc.

Ralescu, A. L., & Fadlalla, A. (1990). The issue of semantic distance in knowledge representation with conceptual graphs. Proc. Of AWOCS90, 141–142.

Rekimoto, J. (1999). Time-machine computing: a time-centric approach for the information environment. Proceedings of the 12th annual ACM symposium on User interface software and technology (p. 45–54).

Rhodes, B. J. (1997). The wearable remembrance agent: A system for augmented memory. Personal and Ubiquitous Computing, 1(4), 218–224.

Ryan, N. S., Pascoe, J., & Morse, D. R. (1998). Enhanced reality fieldwork: the context-aware archaeological assistant. Computer applications in archaeology.

Schilit, B., Adams, N., & Want, R. (2008). Context-aware computing applications. Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on (p. 85–90).

Schilit, B. N., & Theimer, M. M. (1994). Disseminating active map information to mobile hosts. Network, IEEE, 8(5), 22-32. doi:10.1109/65.313011

Schmidt, A. and Aidoo, K. and Takaluoma, A. and Tuomela, U. and Van Laerhoven, K. and Van de Velde, W. (1999). Advanced Interaction in Context. HandHeld and Ubiquitous Computing, Lecture Notes in Computer Science, 1707/1999, 89-101. doi:10.1007/3-540-48157-5_10

Schmidt, A., Takaluoma, A., & M\äntyj\ärvi, J. (2000). Context-aware telephony over WAP. Personal and Ubiquitous Computing, 4(4), 225–229.

Sowa, J. F. (1999). Conceptual graph standard. http://users.bestweb.net/~sowa/cg/cgstand.htm.

Sowa, J. F. (2000). Knowledge representation: logical, philosophical, and computational foundations (Vol. 594). MIT Press.

Sowa, J. (1984). Conceptual structures: information processing in mind and machine. Reading Mass.: Addison-Wesley.

Strang, T., & Linnhoff-Popien, C. (2004). A context modeling survey. Workshop on Advanced Context Modelling, Reasoning and Management as part of UbiComp.

Van Harmelen, F., Lifschitz, V., & Porter, B. (2008). Handbook of knowledge representation. Elsevier Science Ltd.

Want, R., Hopper, A., Falcao, V., & Gibbons, J. (1992). The active badge location system. ACM Transactions on Information Systems (TOIS), 10(1), 91–102.

Weiser, M., & Brown, J. S. (1996). The coming age of calm technology [1]. Xerox PARC. Retrieved July, 8, 2007.

Zhong, J., Zhu, H., Li, J., & Yu, Y. (2002). Conceptual graph matching for semantic search. Conceptual Structures: Integration and Interfaces, 92–106.