2005

Change and planning in chance discovery.

Zhiwen Wu
University of Windsor

Change and Planning in Chance Discovery

by

ZHIWEN WU

A Thesis

Submitted to the Faculty of Graduate Studies and Research

through the School of Computer Science

in Partial Fulfillment of the Requirements for the Degree of Master of Science

at the University of Windsor

Windsor, Ontario, Canada

2005

Your file Votre référence
ISBN: 0-494-09848-1
Our file Notre référence
ISBN: 0-494-09848-1

Abstract

The discovery of risks and opportunities, known collectively as chances, can have a significant impact on decision making. Chances (risks or opportunities) can be discovered from our daily observations and background knowledge. A person can easily identify chances in a news article. In doing so, the person combines the new information in the article with some background knowledge. Hence, we develop a deductive system to discover relative chances with respect to a particular chance seeker.

A chance discovery system that uses a general purpose knowledge base and specialized reasoning algorithms is proposed. The thesis evaluates the implementation of this chance discovery system and discusses the achievements and limitations of its elements, such as Natural Language Processing Tool, Knowledge Entry Tool, Inference Engine and Planner. Finally, A case study about a virtual transportation planning domain implemented using SHOP planner is presented. Example chances are detected in this domain.

Dedication

To my parents, Fawei wu and shaoqun chen.

Without their support and guidance,

I would not survive and be the best to reach my goals...

To my sister, Xiuyun Wu.

God bless her and her family

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 What is Chance Discovery?

According to [40], a *chance* is a piece of information about an event or a situation with significant impact on decision-making of humans, agents, and robots. A *chance* is also a suitable time or occasion to do something. A *chance* may be either positive -an opportunity or negative -a risk. The *discovery* of a *chance* is to become aware of it and to explain its significance. We promote desirable effects when a *chance* is a positive opportunity. On the other hand, we prepare preventive measures in case of a risk. For example, predicting a looming earthquake represents a *chance discovery.*

1.2 Philosophical Considerations in Chance Discovery

1.2.1 Open System

Prendinger and Ishizuka [44] propose that chance discovery best applies to open systems [47]. Open systems are abstractly modelled by cybernetics [6] and systems theory [7]. Examples of open systems include 'living' systems (such as human beings), scientific communities, companies, and artificial systems. There are two essential mechanisms in open system:

- *Regulatory* mechanism of open systems can actively influence the evolution of the system and continuously counteract influences that move the system away from its ideal state.

- *Anticipation* mechanism is to deal with the complexities and influences of the environment. *Anticipation* focuses on the creation of possible and desirable futures, and plans to bring them.

The important feature of open system which makes chance discovery possible is that the future is uncertain but is possible to influence.

1.2.2 Metacognition

In [35], Oehlmann argues that metacognition [9, 14, 16] is a necessary requirement in chance discovery. Metacognition refers to the active monitoring and consequent regulation of the agent's own cognitive processes. It can be simply defined as "thinking about thinking" and is a term in educational psychology referring to achieving successful learning through strongly metacognitive controls, including planning how to approach a given learning task, monitoring comprehension and evaluating progress toward the completion of a task. Knowledge about cognition and the self-regulating mechanisms are necessary elements for chance discovery.

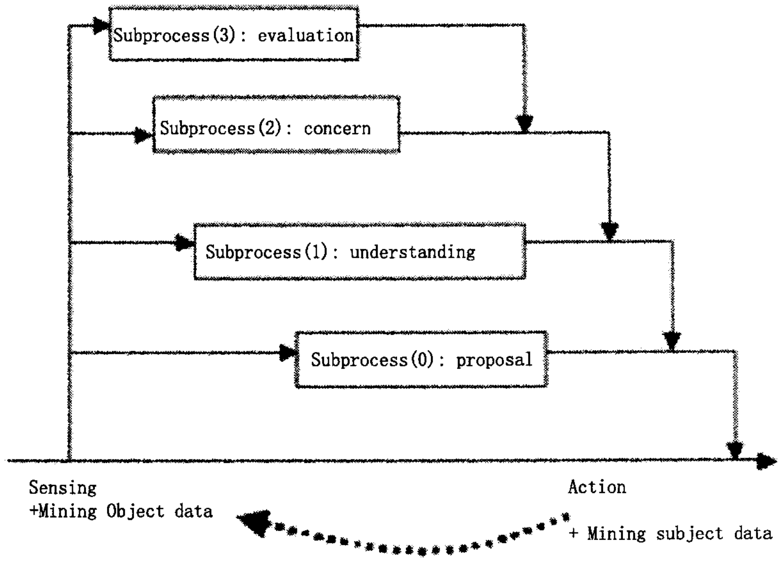

Figure 1.1: The Subsumption Model of Chance Discovery [8]

1.2.3 Model of Chance Discovery Process

Figure 1.1 shows the subsumption model of chance discovery which was first proposed by Brooks [8]. This model is similar to a robot's decision process. A robot (chance seeker) observes some information of the world, evaluates the information, understands the information, makes some proposals of actions according to the information, finally performs a real action which is supposed to be taking advantage of the information (possible chances). Then new actions move the world to a different state. Afterwards, new observations become available as the system starts a new cycle.

1.3 Knowledge Discovery from Data Versus Chance Discovery

As a new research field, chance discovery (CD) has attracted considerable interest. CD is usually confused with the more mature concept of Knowledge Discovery from Data (KDD). In the following section, we discuss the difference between KDD and CD.

KDD tries to discover most likely patterns in data and assumes that these patterns will continue to be valid. However, CD tries to find data that represent chances. It may not match most likely trends but indicate interesting phenomena which were not exploited before and may lead to potential future trends.

The discovery goal in KDD can be divided into a *descriptive* and a *predictive* aspect [44]. The *descriptive* aspect is that the system searches for patterns (or models) in data. The *predictive* aspect is that the system predicts the future behavior of some entities by looking for similar patterns. There are many established data mining methods to achieve those goals, such as classification, regression, clustering, summarization, dependency modeling, and deviation detection [11, 57].

CD may use the knowledge extracted by data mining methods. However, existing data-analysis and data-mining methods cannot explain the significance of unobserved features not described in the given data, or rare events people seldom count as worthwhile predicting. For example, humans that are infected with plasmodium vivax are very likely to contract malaria. However, some people do not. In KDD, those people may be ignored since they do not follow the most likely case. However, in CD, the explanation of their resistance to malaria represents a chance for a significant scientific discovery.

The prevalent process of KDD [12] is shown in Figure 1.2. Raw data are transformed into representations that can be mined. Data mining techniques are then applied to the data resulting in the discovery of event relations, i.e. patterns. The patterns are then evaluated and interpreted by human beings to produce new knowledge.

Ohsawa [38] proposed a process for CD shown in Figure 1.3. In this process, the outputs of computers are evaluated and interpreted. These explanations serve as clues or chance candidates . These clues and candidates stimulate human's thinking and help them focus on certain clues or candidates. This subset of clues becomes input again while others are discarded. Two cycles are formed. One cycle represents the computer analysis process. The other represents the human's thinking process. Knowledge flows from one cycle to the other. Chance is discovered at some point of human's thinking process. As we can see from Figure 1.3, CD focuses strongly on the interaction between human and computers.

In section 1.1, we stated that predicting a looming earthquake represents a "chance discovery". To illustrate the difference between data mining and chance discovery, we consider the following cases of methods leading to this prediction.

First, there are some phenomena that usually precede an earthquake, such as large groups of animals migrating, volcano activities increasing etc.. When another earthquake occurs, all these phenomena may occur similarly. In other words, the phenomena form patterns or

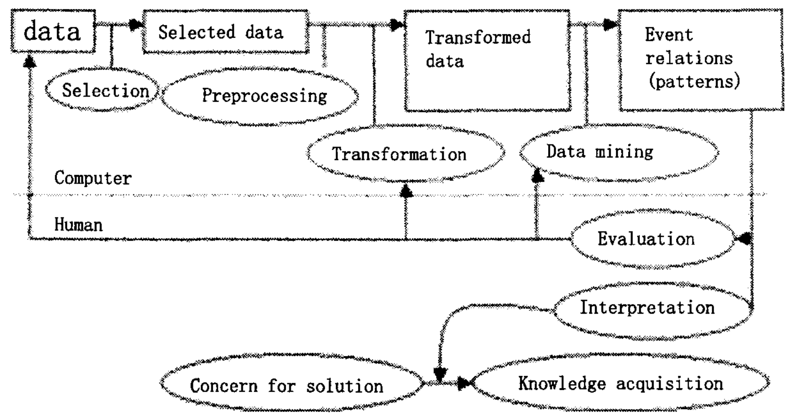

Figure 1.2: Fayyad's Model of the Knowledge-Discovery Process [12]

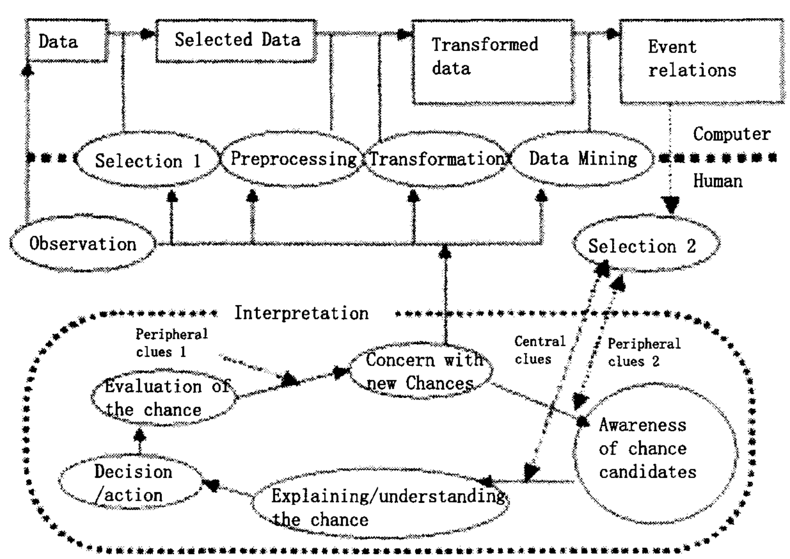

Figure 1.3: Ohsawa's Process of Chance Discovery [38]

models. If we see the same patterns in the future, we may conclude that an earthquake is looming. Patterns (knowledge) are obtained by mining the vast previous observations (data). This method is essentially data mining.

The second method can be regarded as a modified data mining technique. The basic idea is that if rare events known to co-occur with important events (known patterns/models) may form a new part of patterns. For example, recent observations show that, besides the well known phenomena mentioned above, certain bird X always shouts strangely before an earthquake occurs. The phenomenon of X may only co-occurs with those known phenomena several times. In traditional data mining, X may be regarded as a noise. However, if X is observed in a city with no volcano around and no large group of animals, it's reasonable to predict that an earthquake is approaching.

The third method is reasoning, It would be ideal if we could precisely predict an earthquake by exactly calculating the motion of the earth's crust. Many approaches have been applied to this kind of prediction, such as game theory, abductive analogical reasoning, etc. The second and third methods are chance discovery methods. Chapter 2 will discusses them in details.

1.4 Motivation & Objective

This thesis introduces a new approach of chance discovery. We focus on informing the right person the right known information at the right time. We evaluate why a piece of information is important to a particular chance seeker at a particular time by calculating the possible consequences in a virtual reality/model. For example, we know the rule that people may die in an earthquake. If a terrible earthquake looming on an isolated island in the Pacific ocean is correctly detected, most people except some scientists may not pay much attention to this information. However, local native tribes can save their lives by knowing this information. In modern society, we are in a sea of information. Timely disseminating the right information is important to us.

Incorporating the existing chance discovery theory and the aspects we proposed, we aim to achieve the following:

- Demonstrate that chance discovery is a process that tries to identify possibly important consequences of information with respect to a particular person or organization at a particular time; Chance can be discovered from changes that are reported in text documents around the world; A chance discovery system, which is knowledge based and can evaluate change from the point of view of logic, is a viable technique for chance discovery.

- Present the theoretical aspects of the knowledge-based chance discovery system.

- Present and discuss the implementation of this chance discovery system.

1.5 Contribution

- A new knowledge-based framework and architecture for chance discovery [58].

- An implementation that shows the practicality of our system. It also demonstrates the features of our system compared to the existing approaches, as well as its limitations.

1.6 Outline of the Thesis

In this chapter, we introduced the concept of chance discovery and the difference between knowledge discovery from data and chance discovery. We also presented the motivation and contribution of this thesis. Chapter 2 introduces the existing approaches for chance discovery. Chapter 3 presents our knowledge-based chance discovery system. In Chapter 4, we discuss the implementation of our chance discovery system. Chapter 5 concludes the thesis and discusses some future work.

Chapter 2

Necessary Roles in Chance Discovery

As the personal computer pioneer Alan Kay said: "The best way to predict the future is to invent it". An essential aspect of a chance is that it can be a new seed of significant future. To make inventing the significant future possible, we need timely identification of the chances as well as drawing humans' attention to such chances and to knowledge for dealing with them. Ohsawa [38] proposes three roles in chance discovery: Communication, Imagination and Data Mining. Many approaches have been proposed around these roles and show how these roles contribute to chance discovery.

2.1 Communications

The emergence of new ideas in human-human, human-agent, and agent-agent communications in virtual or real communities may activate the process of chance discovery.

2.1.1 Argumentation-Based Chance Discovery

McBurney and Parsons [30] propose an argumentation based communication approach for chance discovery in domains that have multi-agents. Each agent may only have a partial view of a problem and may have insufficient knowledge to prove particular hypotheses individually. For example, a river flows across several regions. If we have several computer agents in each region, each agent only knows the weather in its own region over a period of time. If these agents talk with each other and report rain in their own regions, then a flood may be predicted.

Mcburney and Parsons [30] defines locutions and rules for this type of dialogues, called *discovery dialogues.* It is not one of the conventional six types. According to Walton and Krabbe [54], the six primary types of dialogue are: Information-seeking dialogues, Inquiry dialogues, Persuasion dialogues, Negotiation dialogues, Deliberation dialogues, Eristic dialogues.

Discovery dialogues intend to discover something not previously known. It means that discovery may only emerge in the course of the dialogue. Like most actual dialogues, *discovery dialogues* involve mixtures of those six dialogue types. *Discovery dialogues* may be viewed as

> *Inquiry dialogues*– The participants collaborate to answer some question or questions whose answers are not known to any one participant.

However, they are not disinterested in searching for truth like *inquiries. Discovery dialogues* may be only interested in chances. For example, a discussion about possible risks of some system triggers a search not for all possible outcomes, but only for those with negative consequences.

When a chance is discovered, *discovery dialogues* are about what to do to prevent, or enhance a chance event. In this case, *discovery dialogues* may be viewed as

> *Deliberation dialogues–* Collaborate to decide what action or course of action should be adopted in some situation.

Discovery dialogues need to be automated. A communication language and a set of protocols are necessary for computational entities involved. In [32], a set of five requirements for the design of languages and protocols for communications in chance discovery is derived as follows.

- The language should transmit domain-specific information in an appropriate form between participants.
- The participants have the ability to argue with each other concerning the messages they transmit.
- The communications protocol used by the agents must encode some theory of debate or argument, what is called a logic of argumentation.
- Communications language should enable the participating agents to articulate relevant changes in their internal states, i.e., the expression of self-tranformation [15, 33].
- It can enable an appropriate mechanism for resolution of differences of beliefs or intentions of the participants.

Generic Languages and *dialogue game protocols* are two proposals used in multi-agent systems community for agent communications [32]. FIPA ACL [13] stands for the Foundation for Intelligent Physical Agents' Agent Communications Language. However, FIPA ACL is unsuitable as a protocol to support chance discovery because FIPA ACL doesn't support *Inquiries* and *deliberations.* According to the five requirements, *Dialogue game protocols* [31] have greater potential capability to support chance discovery between autonomous agents, An example of formal dialogue Game protocols is described in [31].

2.1.2 Concept Articulation

A good online shopping recommender system can exploit concept articulation technique to be more profitable. Today's society is a society with oversupply of merchandise. A product which only has the necessary functions is not enough to satisfy customers. However, the same ordinary product can be sold at a high price by convincing customers that it matches their taste. Shoji and Hori [49, 48] propose that creative communications in real shopping behavior can lead to a customer's mental leap. The appropriate information provided by the clerk at the right time can change customers' focus and lead to a successful sale.

Purchasing can be roughly divided into a *problem-solving* type and a *concept-articulation* type. *Concept-articulation* applies when customers do not have clear requirements. Their requirements get gradually clearer as information is provided as they examine various products and talk with the sales clerks. Sales clerk's communication patterns could be classified as *expected reaction* or *unexpected reaction.*

An *expected reaction* is to show customers what they want. *Unexpected reactions* present information from a different viewpoint than the customers' current thought. Let's see the following examples:

- **Example 1. Expected reaction**:
  - *Customer*: This cheese burger is too large for me. Do you have a small one?
  - *Sales clerk*: (smiling) How about this one (small burger)? But it is a garlic burger.
  - *Customer*: Well, let me see ... Sorry, I got to go.
- **Example 2. Unexpected reaction**:
  - *Customer*: This cheese burger is too large for me. Do you have a small one?
  - *Sales clerk*: Cheese burger is the best burger in our restaurant. Our cheese burger is made from special low fat cheese. It just looks large but you will feel nothing.
  - *Customer*: Oh, great! ... .

Example 1 leads to an unsuccessful sale and example 2 leads to a successful one. Figure 2.1 shows that, in case of the *unexpected reaction* (example 2), a new focus (low fat) presented by the sales clerk triggers the customer's mental leap. The attitude of the customer about large cheese burger changes from 'too large' to 'excellent choice'.

In this kind of communication, the customer may already have some opinions about what his or her needs. But the product (potential chance) is not exactly like what he expects. The customer presents

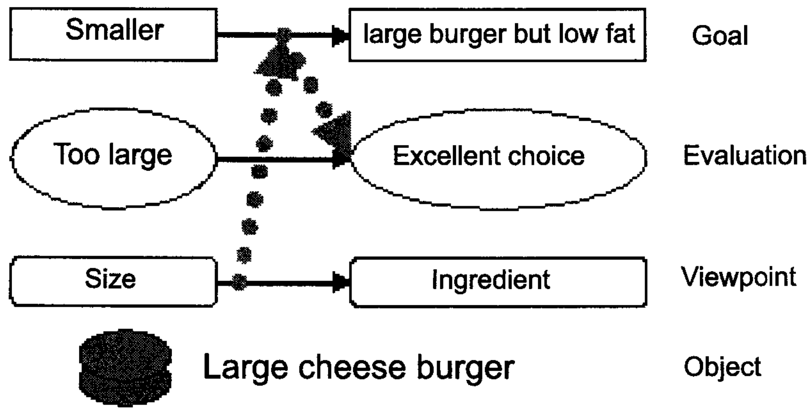

Figure 2.1: Changes Caused by the Unexpected Reaction

his comments about the product. By evaluating the comments, the *unexpected reaction* can change the customer's focus and convince the customer to accept some new viewpoints smoothly. According to this, Shoji and Hori [50] develope a system called S-Conart (Concept Articulator for Shoppers) to support this type of communications.

2.1.3 Influence Diffusion Model

In text-based communications, such as email, on-line forum, ... etc., people would join the discussion if a topic or comment sounds interesting to them. Matsumura [22] proposes the influence diffusion model (IDM) to find valuable information such as influential comments, opinion leaders, and interesting terms form the archives of text-based communications. For example, if C_y replies to C_x, the influence of C_x onto C_y, $i_{x,y}$, is defined as

$$i_{x,y} = \frac{|w_x \bigcap w_y|}{|w_y|} \tag{2.1}$$

where $|w_y|$ denotes the count of terms in C_y and $|w_x \bigcap w_y|$ denotes the count of propagated terms from C_x and C_y. If C_z replies to C_y, the influence of C_x onto C_z via C_y, $i_{x,z}$ is defined as

$$i_{x,z} = \frac{|w_x \bigcap w_y \bigcap w_z|}{|w_z|} \cdot i_{x,y} \tag{2.2}$$

There are three models included in the IDM [22]:

1. IDM for comments, the influence of a comment C_i (D_{c_i}) can be described as

   $i_{i,r} = \frac{|w_i \bigcap w_j \bigcap \ldots \bigcap w_q \bigcap w_r|}{|w_r|} \cdot i_{i,q}$

   $I_{\S_i} = i_{i,j} + i_{i,k} + \cdots + i_{i,y} + i_{i,z}$

   $D_{C_i} = \sum_{\S_i \in E_i} I_{\S_i}$

   $\S_i$ is the comment chain which starts from C_i, $I_{\S_i}$ represent the sum of influences diffused from C_i in $\S_i$, E_i is the set of all comments chains that start from C_i.

2. IDM for participants.

   The influence of participant P will be the sum of the influence of P's comments. Let D_p be the influence of P, and k_p be a collection of comments posted by P. Then, D_p is described as

   $D_p = \sum_{i \in k_p} D_{C_i}$

3. IDM for terms.

   The influence of term t (D_t) can be described as :

   $j_{i,r,t} = \frac{1}{|w_i \bigcap w_j \bigcap \ldots \bigcap w_r|} |w_r| \cdot i_{i,r},$

$$J_{\xi_i,t} = j_{i,j,t} + j_{i,k,t} + ... + j_{i,y,t} + j_{i,z,t}$$

$$D_t = \sum_{\xi_i \in E_t} J_{\xi_i,t}$$

E_t represents the set of comment's chains. Each comment in a chain contains term t.

This algorithm can efficiently identify valuable topics or terms in text-based communications. However, It does not evaluate logical relationships. Sometimes it may lead to meaningless findings. For example, a virus may copy the topic and keep replying to the same topic by pasting the same sentence.

2.1.4 Other Communication Approaches

Sumi and Mase [51] introduce two systems for enhancing daily conversation that can increase opportunities to encounter new ideas and future partners for collaboration. One is Augmented Informative Discussion Environment (AIDE). AIDE enhances online discussion with visualization of the discussion structure and a virtual discussant. The users of AIDE can mutually notice the similarity and difference among their viewpoints with respect to common topics. The other is AgentSalon which enhances casual face-to-face chatting.

McArthur and Bruza [28, 29] develop a method for collecting and storing the utterances: a vector representation of words representing aspects of both presemantic and semantic context. They derived post-semantic context based on vector representations of words and pursued the vector representation to discover *ebbs* and *flows* of socio-cognitive 'meaning' within a online community and between communities. Perry [43] gives the distinction among presemantic, semantic and postsemantic context. The presemantic context is what gives an ambiguous linguistic meaning, for example, in the sentence "I saw her duck under the table", whether "her" is a pronoun or an adjective is not clear. The postsemantic context represents some tacit knowledge. For example, John says: "It is raining". We assume that it is raining in John's location.

2.2 Imagination

In the process of chance discovery, it is important to involve oneself in a new context where the current chance can be significant, i.e., human's perception of chance. This might be an analogical matching of one's own experience with the current rare situation, or an imagination of a scene or a story in the future situation. This subsection will investigates the theories and approaches for imagination in chance discovery.

2.2.1 Bayesian Chance Discovery

In [52], Tawfik proposes that chance discovery represents a dilemma for inductive reasoning from a reasoning point of view. The Bayesian approaches represent a potential solution. Traditionally, forecasting has relied on extrapolation. Extrapolation is a form of inductive reasoning that assumes that current trends would carry on into the future. Finding the proper knowledge representation is of great importance for chance discovery. Conventional knowledge representation and reasoning frameworks will be likely to miss rare events because they favor "normal" and "common", rather than "rare" and "exceptional". The following paradox shows an example of the dilemma of induction.

> The color of an emerald is "grue"(green then blue) if it is and has always been observed green until some future time (say year 2222) when it will turn blue. This notion presents a paradox to inductive reasoning because our observations support the statement that emeralds are green as well as the claim that they are grue (Goodman 1955).

The problem is caused by the inductive assumption which implies the future will look like the past.

The Bayesian approach is to explicitly explore all possibilities including all alternative models (all possible worlds), assess priors and conditional probabilities, and calculate posterior probabilities given all available observations for the different models under consideration. The probability of a statement S given some evidence E is given by (h_j represents all possible models consistent with E):

$$P(S|E) = \frac{P(S)P(E|S)}{\sum_j P(E|h_j)P(h_j)} \tag{2.3}$$

Tawfik suggests that a knowledge representation suitable for chance discovery should be able to concisely encode a possibly very large number of models (possible worlds). The chance discovery problem can be represented by a Kripke structure [19].

$$M = (W, \phi, \pi, R) \tag{2.4}$$

W denotes a set of worlds. Each world is described using truth assignment π defined for a set of propostions ϕ. An accessibility relation R determines the set of worlds reachable from a particular world. Each world w ($\in$ W) occurs with a probability $\mu(w)$. The probability of a proposition φ is given by

$$P(\varphi) = \sum_{w \models \varphi} \mu(w) \tag{2.5}$$

We can evaluate whether φ is a chance or not based on $P(\varphi)$.

This deductive approach to chance discovery has some complexity and feasibility limitations. It is difficult to encode all possible combinations of events, actions and assumptions and their consequences. Therefore, *Backward* chaining may be applied to achieve a more efficient solution.

2.2.2 Abductive and Analogical Reasoning

Abductive Reasoning

Peirce [42] characterized the definitions of *abductive* and *inductive* reasoning as:

> *abductive* reasoning is an operation for adopting an explanatory hypothesis, which is subject to certain conditions, and that in pure abduction, there can never be justification for accepting the hypothesis, other than through interrogation. *Inductive* reasoning is an operation for testing a hypothesis by experiment.

The following example shows the difference between *inductive* and *abductive* reasoning. When we find fossils of sea shells in Tibet highland, the conclusion form induction is that more fossils of sea shells will be found if we dig there. On the other hand, the result from abduction is that Tibet highland used to be under the sea.

The following formula shows how *abductive* reasoning works.

F is a set of facts,

H explains F (H is a hypothesis),

No other hypothesis explains F as well as H does.

Therefore, H is probably correct.

For example, since aliens can explain UFO, we conclude that there are aliens from outer space.

Analogical Reasoning

Analogical reasoning (analogy) [17] is a widely used reasoning technique. There are two main components of analogy: *source* and *target.* *Source* refers the knowledge we are familiar with and *Target* is the knowledge we attempt to explain. Reasoning is achieved by analogical mapping from *source* to *target.* For example, if we know that human beings die someday, we can guess that aliens may die in the same way as human beings in outer space.

Russell [45] proposed DBAR (determination-based analogical reasoning). The determination rule (P determines Q) is as follows:

$$P(x, y) \succ Q(x, y) \text{ iff}$$
$$\forall wyz[P(w, y) \bigwedge Q(w, z) \Rightarrow \forall x[P(x, y) \Rightarrow Q(x, z)]]$$

where x is a set of variables such as a, b, c, ... Then, the analogical mapping can be shown as follows:

$$\frac{(P(x, y) \bigwedge Q(x, z)), P(S, A), P(T, A), Q(S, B)}{Q(T, B)} \tag{2.6}$$

The following shows an example of analogical reasoning.

species(Penguin,bird), colour(Penguin,black-and-white)...

species(dove,bird), instinct(dove,fly), ...

.....

We have a determination rule as

$species(x, n) \succ instinct(x, l)$,

By using the determination rule and knowledge regarding Penguin's source knowledge, we can infer

instinct(Penguin, fly).

Abductive Analogical Reasoning

Abduction requires a complete hypotheses set which contains all the necessary hypotheses. To explain an observation, a consistent hypotheses subset is selected. In [3, 4], ABE proposes that a chance can be regarded as an unknown hypothesis. The conventional hypothetical reasoning system cannot generate new hypotheses. *Abductive Analogical Reasoning* (AAR) [2, 1] combining *abduction* and *analogical mapping* is an extension of hypothetical reasoning. AAR can lead to chance discovery because it can generate new hypotheses (the missing knowledge by imagination). The following shows how AAR generate missing hypothesis to explain an observation.

Definition 1: Analogical clause: A is an analogical clause of B means that B is analogically mapped from A.

Definition 2: Analogical Transform: Let A be a set of clauses, A |>> A' represents A' is analogically transformed from A. i.e., A' is a set of clauses that can derive analogical clauses of the clauses derived from A.

O is an observation, AAR tries to explain it using clauses in the knowledge base $\triangle$. But

$$\triangle \nvDash \text{O (O cannot be explained by only } \triangle)$$

Then, AAR returns a set of minimal clauses S such that

$$\triangle \vDash S \bigvee O \text{ (} S \bigvee O \text{ can be explained by } \triangle \text{),}$$
$$\triangle \nvDash S \text{ (S cannot be explained by } \triangle).$$

Therefore, $\neg$S is a missing clause set that is necessary to explain O since ($\triangle \nvDash S$) and ($\neg S \nsubseteq \triangle$). We find clause set S'. S' satisfies

$$\text{S} \mid >> \text{S' (S' is } \textit{analogically transformed} \text{ from S),}$$
$$\triangle \vDash S' \text{ (S' can be explained by } \triangle \text{).}$$

If we perform another analogically transform, we get S" as follows,

$$\text{S'} \mid >> \text{S"},$$
$$\triangle \vDash \text{S"} \bigvee O,$$
$$\triangle \nvDash \text{S"}.$$

Then, $\neg$S" is a newly generated hypotheses clause set and can explain observation O. In AAR, A chance is regarded as a set of abductive hypotheses by performing AAR from the possible future observation.

2.2.3 Representational Change and Packing

Dietrich et al [10] proposed that *representational change* in humans and machines is a great source for chance discovery because it produces clues triggering chances. Dietrich gives the following example of *representation change.*

water **flows** *from a place with greater pressure to a place with less pressure.*

If we change 'flows' to 'move',

water **moves** *from a place with greater pressure to a place with less pressure.*

We know Planets, feet and money move. When 'flow' is *abstracted* to 'move', this rule can be applied to more objects.

Packing is another process of representational change by suppressing irrelevant information in a structured representation. This suppression alters the structure of the representation by making irrelevant information between base and target less accessible. For example, if we consider two strings 'abab' and 'efghefgh', they looks different to each other. However, They are clearly similar since they both have the structure 'xx' (repeating sequences). By packing irrelevant information, we can apply a well known world's rule to an unknown world and discover chance from the unknown world. However, we need good packing algorithms.

2.3 Data Mining

Data Mining techniques in chance discovery are different from the conventional ones since we want to discover something different from the trend. Recent research shows that analysis on network structure of data has some promising results.

2.3.1 KeyGraph

KeyGraph [39] is a text indexing method which extracts keywords that represent the main point of a document without relying on external devices such as document corpus or other natural language processing tools. KeyGraph can find important but rare keywords from a document. The basic idea is that if a keyword links two basic keyword clusters (set of keywords with high frequency) together, it is regarded as important keyword even if its frequency is low.

Ohsawa outlines the KeyGraph algorithm by regarding a document as a building:

> A building (document) has foundations (statements for preparing basic concepts),walls, doors and windows(ornamentation). The roofs (main ideas in the document), which protect the building's inhabitants against rains or sunshine, are considered to be the most important. These roofs are supported by columns. KeyGraph algorithm finds the roofs.

The processes of KeyGraph are composed of four phases [39]:

1. Document(D) preparation:

   Stop words which have little meaning are removed from D. Words and phrases are stemmed and identified.

2. Extracting foundations from D:

   A graph G for document D is constructed of nodes representing terms, and links representing the co-occurrence (term-pairs which frequently occur in same sentences throughout D). Nodes and links in G are defined as follow:

Nodes– Nodes in G represent high-frequency (HF) terms in D because terms might appear frequently for expressing typical basic concept.

Links–Nodes in HF are linked if the association between the corresponding terms is strong. The association of terms W_i and W_j in D are defined as

$$assoc(W_i, W_j) = \sum_{s \epsilon D} min(|W_i|_s, |W_j|_s) \tag{2.7}$$

where $|X|_s$ denotes the count of x in sentence s. Pairs of high-frequency terms in HF are sorted by assoc and the pair above the ((number of nodes in G)- 1) tightest association are represented in G by links between nodes.

3. Extracting columns from D:

   The probability of term w to appear is defined as key(w), and the key(w) is defined by

$$Key(w) = 1 - \prod_{g \subset G}(1 - \frac{\sum_{s \in D} |w|_s |w - g|_s}{\sum_{s \in D} \sum_{w \in s} |w|_s |w - g|_s}) \tag{2.8}$$

   Sorting terms in D by keys produces a list of terms ranked by their association with cluster, and the twelve (an estimated number) top key terms are taken for high key terms.

4. Extracting roofs from D:

   (a) Add all the high key terms as new nodes to G if they are not in G yet

   (b) The strength of column between a high key term W_i and a

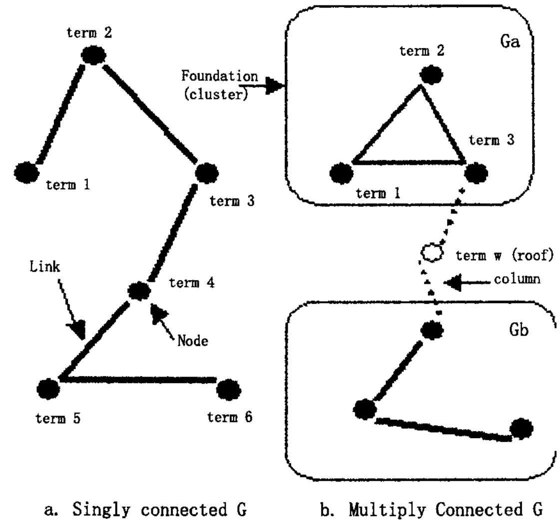

Figure 2.2: An Example Output of KeyGraph [39]

high frequency term W_j, is expressed as

$$columns(W_i, W_j) = \sum_{s \in D} min(|W_i|s, |W_j|s) \qquad (2.9)$$

(c) Columns touching W_i are sorted by $column(W_i, W_j)$. For each high key term W_i, Columns with the highest column values connecting term W_i to two or more clusters are selected to create new links in G representing columns by dotted line.

(d) Finally, nodes in G are sorted by the sum of column of touching columns. Terms represented by nodes of higher values of these sums than a certain threshold are extracted as the keywords for document D.

Figure 2.2 shows an example output of keyGraph. Ga and Gb are foundations (basic keywords). The dotted line is columns. Roof (w) is supported by columns. Term w is considered to be a important keyword (chance).

2.3.2 Small World

Watts and Strogatz [55, 56] introduce a graph called small world which is neither completely regular nor completely random, but have instead a "small world" topology in which nodes are highly clustered yet the path length between them is small. According to Watts and Strogatz, a social graph (e.g. the collaboration graph of actors in feature films), a biological graph (e.g. the neural network of the nematode worm C. elegans), and a manmade graph (e.g. the electrical power grid of the western United States) all have a small world topology. World Wide Web also forms a small world network [5].

The node's contribution to make a graph becoming small world can be a measurement of its importance. Important nodes represent chances.

A small world graph is defined as one in which L $\geq L_{rand}$ (or L $\cong L_{rand}$) and $C \gg C_{rand}$ where L_{rand} and C_{rand} are the *characteristic path length* and *clustering coefficient* of a random graph with the same number of nodes and edges. The *clustering coefficient* and the *characteristic path length* are defined as:

- The *characteristic path length* (L) is the path length averaged over all pairs of nodes. The path length d(i, j) is the number of edges in the shortest path between nodes i and j.
- The *clustering coefficient* (C) is a measure of the cliqueness of the local neighbourhoods. For a node with k neighbours, then at most $k(k-1)/2$ edges can exist between them. The clustering of a node is the fraction of these allowable edges that occur. The *clustering*

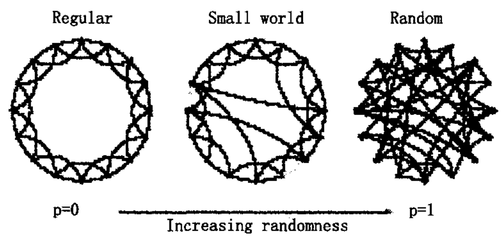

Figure 2.3: Random Rewriting of a Regular Ring Lattice [55]

coefficient is the average clustering over all the nodes in the graph.

Watts and Strogatz propose a model of graph which is called β-Graphs. Starting from a regular graph, they introduce disorder into the graph by randomly rewiring each edge with probability p as shown in Figure 2.3. The graph can be one of the following cases:

- If p = 0 then the graph is completely regular and ordered.
- If p = 1 then the graph is completely random and disordered.
- Graphs with intermediate values of p are neither completely regular nor completely disordered. They are small worlds.

The following is how to construct a small world from text data (for example, an article). This method is similar to keygraph algorithm.

1. Document preparation (the same process as KeyGraph, see section 2.3.1).
2. Frequent terms which appear over a user-given threshold are extracted and fixed as nodes.
3. For every pair of nodes, the co-occurrences are counted. An edge is added if *Jaccard coefficient* [18] exceeds a threshold (J_{thre}). The

Jaccard coefficient is simply the number of sentences that contain both terms divided by the number of sentences that contain either term.

Matsuo et al [27] proposed an algorithm to pick up rare but important terms (chances) by admitting that a document is a small world, as follows:

- Extended path length

$$d'(i,j) = \begin{cases} d(i,j), & \text{if(i,j) connected,} \\ & \text{the shortest path between them;} \\ W_{sum}, & \text{otherwise,} W_{sum} \text{ is a constant} \\ & \text{representing an estimated length.} \end{cases}$$

- Extended characteristic path length (L') is the averaged extended path length over all pairs of nodes.
- L'_v is an extended path length averaged over all pairs of nodes except node v. L'_{G_v} is the extended characteristic path length of the graph without node v.
- The contribution CB_v of the node v to make the world small is defined as $CB_v = L'_{G_v} - L'_v$.

When a node v has a large CB_v though its frequency is low, it was considered to be important. Because this term helps to merge the structure of the document.

Small world is based on the similar idea as KeyGraph: If a node (an event) shares an important position in a graph. It might have an impact even if the frequency of the event is low. In KeyGraph case, the importance is defined as co-occurrence in two or more big clusters. In small world case, contribution in making the graph highly connected is important.

2.3.3 Priming Activation Indexing

Priming Activation indexing (PAI) [25] is another method for extracting the author's main points from a document. PAI is similar to KeyGraph and Small World but employs another criterion for measuring keyword importance: If the flow on the graph is through a certain node, the node is important. The algorithm of PAI consists of five phases:

1) **Pre-processing:** The same as keyGraph, see section 2.3.1.

2) **Segmentation:** A document is segmented into portions S_t (t=1,2,..., n) based on a semantic coherence criterion.

3) **Activation Network:** For each segment S_t, terms are sorted by their frequencies and top 20% terms are denoted by K(t) as fundamental concepts. The association of terms W_i and W_j ($assoc(W_i, W_j)$) is defined in equation 2.7. Pairs of terms in K(t) are sorted by assoc and the pair above the ((number of terms in K(t))- 1)th tightest associations are linked. For links between W_i and W_j, $R(t)_{i,j}$ is defined as

$$R(t)_{ij} = \frac{assoc(W_i, W_j)}{links(W_i)} \tag{2.10}$$

Where links(W_i) denotes the number of links connected to W_i.

4) **Spreading Activation:** From S_1 to S_n, activities are propagated by iterating

$$A(t) = ((1-\gamma)I + \alpha R(t))A(t-1) \tag{2.11}$$

(1)	Input sequence: Event sequence D
(2)	Periods('.'s): The moments of major changes
(3)	Islands: Fundamental set of items, co-occurring frequently
(4)	Bridges: Event-island co-occurrences representing causalities
(5)	Hubs: potentially significant events connecting multiple islands

Table 2.1: Factors in KeyGraph [36]

where A(t) is a vector represents the activities of nodes at discrete step t=1,2 ... n. Primal activity of each term before executing spreading activation is 1. I is an identity matrix. The parameters γ and α depend on the characteristics of documents.

5) **Extract Keywords** After spreading activation on all the segments in turn, highly activated terms are considered as the author's main point.

2.4 Applications

2.4.1 Discover Active Faults from Earthquake-Sequences

Ohsawa [36, 37, 38] show the five major components of KeyGraph and proposed that KeyGraph can be applied to various data, where the components (see table 2.1) correspond to meaningful substances in the target world. The keyGraph-based earthquake-data miner is called Fatal Fault Finder (F^3) is to extract active faults with risks of near-future large earthquakes from earthquake-sequences. F^3 is composed of 3 steps.

- **Get the following input data:**

	time	longitude	latitude	depth	magnitude
1	85 01 01 01:17:52.24	142.804E	42.140N	10.2	2.1
2	85 01 01 01:30:49.92	139.523E	37.441N	158.7	3.1
3	85 01 01 01:52.24.20	146.804E	46.140N	230.2	4.2
...	...	...	...	...	...
N	XXXX	XXXX	XXXX	XXXX	XXXX

Table 2.2: An Example of Earthquake Sequence (Data 2) [38]

- **Data 1:** Land-surface locations of faults, i.e. F=(a,b) for every fault F where a and b represent the two-dimensional location of F.
- **Data 2:** A sequence of earthquakes, where each earthquake E_i is given as ($time_i$, $longitude_i$, $latitude_i$, $depth_i$, $magnitude_i$). ($longitude_i$, $latitude_i$), $depth_i$, and $magnitude_i$ denote the two-dimensional position, the depth, and the magnitude of the ith earthquake respectively. Table 2.2 show an example of data 2.

- **Make D:** The distance from the epicenter x of each earthquake in data 2 to a fault F is computed. Then, F is regarded as the focal fault if it is the nearest to x of all the faults in data 1. D is made as the sequence of these obtained focal faults. Each string in D implies an event. '.' is inserted after each earthquake stronger than M_Θ (a fixed magnitude). Finally we can get D the following:

  `D=123#202#1#84#.76#.216#1#202#84#.249#84#.76#249#...` (2.12)

- **Obtain risky faults:** Obtain the events of the highest values of

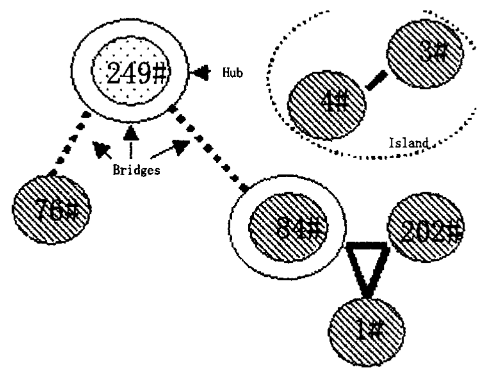

Figure 2.4: An Example Output for Earthquake Sequence [38]

key in D, and regard their focal faults as risky. For example, for Sequence 2.12, 249# is considered risky, see Figure 2.4.

The risky faults that are empirically obtained by keyGraph correspond closely to real earthquake occurrences and seismologists' risk estimation.

2.4.2 Chance Discovery from the WWW

Matsumura et al [26, 24, 23] propose that revealing the structure of the WWW will be a key to understand the real world. As a part of the human social network, the changes in the WWW reflect the development in the real world quickly. A hyperlink is an implicit type of 'endorsement' of the page being pointed to. By analyzing those 'endorsement', we can decode the trends in the human society. The WWW consists of an abundance of communities, each corresponds to clusters of web pages sharing common interests [20]. A web page supported by multiple communities is considered to a popular source which can satisfy wide interests.

The web can be represented by a graph, by representing web pages as nodes and the relation between web pages as links. There are two major relations between a pair of web pages:

1. *Direct relation*: A node represents a web page and a link represents a hyperlink between two web pages.

2. *Co-citation*: A node represents a web page and a link is created by co-citation. co-citation refers to this kind of relation, for example, if page A points to both page B and page C, B and C may be related.

Web pages in the same community may not frequently refer to one another because of competitive relation. For example, an online shop may not have a link which points to its competitors. Therefore, *co-citation* is better than *direct relation* in reflecting the real world.

By doing analogical mapping, KeyGraph can be applied to web pages. Table 2.3 shows the analogy between a document and a Web-page set. Based on this analogy, a web-page set is considered as a document and can be applied to KeyGraph. Figure 2.5 shows an example output of KeyGraph. *Page 8* is considered to be a newly growing important site. Matsumura [24] applied this algorithm to the set of 500 popular web pages obtained by searching Google using the input query 'human genome' and discoverd www.celera.com as an growing important web site in the human genome research domain.

Case of a document:

(1)	Actions of the author, i.e. writing words
(2)	Periods ('.'s): the end of sentences
(3)	basic concepts for the author
(4)	the flow of content, connecting basic concepts and assertions
(5)	the relations of asserted words to basic concepts

Case of a Web-page set

(1)	web pages sequences
(2)	the end of each web page
(3)	most contributed or fundamental web page (website) about certain topic
(4)	The relationship between web pages
(5)	New growing web page (website)

Table 2.3: The Analogy between a Document and a Web-page Set [38]

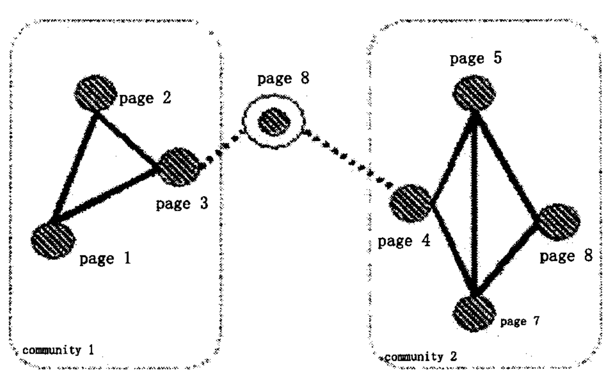

Figure 2.5: An Example Output of KeyGraph for Web-page Set [24]

Chapter 3

Knowledge-based Chance Discovery System

In this chapter, we propose a new architecture for chance discovery. The new architecture has implications to both the conception and discovery of chances that can be summarized as follows:

- Chances are not necessarily unknown hypotheses. As we stated in Chapter 2, some approaches focus on generating new unknown hypotheses and regarding them as chances. However, many chances result from known events and rules. For example, applying for the right job at the right time represents a chance for an employment seeker as well as the employer. In this case, the goal is clear. However, chance discovery means that the employment seeker applies at the proper time and for the employer, it means to correctly project which applicant will be better for the job.

- Inherently, chance discovery has a temporal reasoning component. New risks and opportunities are typically associated with change. An invention, a new legislation, or a change in weather patterns may result in new chances. Incorporating chance discovery in a belief update process is fundamental to this work.

- Chances are relative; someone's trash may be another's treasure. For example, finding a cure for a fatal disease represents more of a chance to an individual suffering from this condition or at risk to contact it.

- To discover chances and take advantage of them, a system which can perform deductive reasoning is needed.

Therefore, we consider chance discovery as a process that tries to identify possibly important consequences of change with respect to a particular person or organization at a particular time. For this to happen, a logical reasoning system that continuously updates its knowledge base, including its private model of chance seekers (CS) is necessary. A chance discovery process may act as an advisor who asks relevant "what if" question in response to a change and present significant consequences much like seasoned parents advise their children. Such advice incorporates knowledge about the chance seekers, their capabilities, and preferences along with knowledge about the world and how it changes.

In a word, to discover chances, we need the followings:

- First, a knowledge base which can infer and understand commonsense knowledge and that can incorporate a model of the chance seeker.

- Second, we need a source for information about change in the world. Information about changes occurring in the world is usually documented in natural languages. For example, a newspaper can serve as a source for information about change. We need nature language processing (NLP) tools to understand this newspaper.

- Third, we need a temporal projection system that would combine information about change with the background knowledge and that would assess the magnitude of the change with respect to the knowledge seeker.

This thesis proposes an approach for assessing the implications of change to the chance seeker and bringing to the attention of the chance seeker significant risks or opportunities.

3.1 Chance and Change

Chance and change exist everywhere in our daily life. In general, changes are partially observable by a small subset of agents. Therefore, it is more likely to learn about changes happening in the world through others. For example, information about change could be deduced from conversations in chat rooms, newspapers, e-mail, news on the WWW, TV programs, new books and magazines, etc. In other words, change causing events occur daily around the world. The amount and rate of those events is very large. However, a relatively small portion of these changes represent risks or opportunities to any particular chance seeker.

Initially, the system starts with a stable knowledge base KB. The knowledge base represents the set of widely held knowledge. As part of KB's knowledge, each chance seeker maintains its own private knowledge that describes its current attributes. In addition to the KB, each chance seeker also maintains its private goals and plans leading to some of the goals. If the chance seeker does not specify a set of goals, the system will use default goals that are widely accepted as common goals. For example, the system assumes that all people want to become more famous or richer, want their family members and relatives to be rich and healthy, etc. We assume that the chance seeker has already exploited the chances present in the current KB and that the current plans of chance seeker are the best according to current KB. However, current plans may only be able to achieve part of the goals. For example, the goal to own a house in Mars is unachieved by current knowledge.

A chance seeker's goal can be represented by a set of sentences describing a future status of chance seeker's attributes. For example, if chance seeker set up the goal to be a famous scientist, the system can judge the achievement of the goal by measuring chance seeker's current attributes, such as education, occupation, published papers, social class, etc. The system maintains an attribute framework for the chance seeker in KB. The attribute framework changes as if necessary. A goal can be considered as a future projection of current framework. On the other hand, a future set of attributes could satisfy many goals of chance seeker. Current plans of the chance seeker project current set of attributes to an achievable set of goals.

As new information B becomes available, an update operation is triggered. The update operation proceeds in two phases: a *explanation* phase and an *projection* phase. The *explanation* phase tries to revise current beliefs that may have been proven to be inaccurate by the occurrence of B. Similarly, the *projection* phase, revises current beliefs to take into account the occurrence of B. A risk is detected if the occurrence of B results in a threat to the causal support for one of the plans of the chance seeker. An opportunity is detected if B satisfies one of the followings: the occurrence of B makes more of the chance seeker's goals achievable, or better plans can come up after B. In some cases, a particular piece of new information will result in both risks and opportunities.

3.2 Cyc Knowledge Base for Chance Discovery

Cyc knowledge base [41] is supposed to become the world's largest and most complete general knowledge base and commonsense reasoning engine and therefore represents a good candidate as a source for background knowledge. The Cyc knowledge base (KB) can be regarded as a formal system containing a vast quantity of fundamental human knowledge: facts, rules of thumb, heuristics, and a reasoning system about objects and events of everyday life by using its own knowledge. The medium of representation is a formal language known as CycL [41]. CycL is essentially an augmentation of first-order predicate calculus (FOPC), with extensions to handle equality, default reasoning, skolemization, and some second-order features. The following shows an example of CycL:

```
(#$forAll ?PERSON1
(#$implies
(#$isa ?PERSON1 #$Person)
(#$thereExists ?PERSON2
```

```
(#$and
(#$isa ?PERSON2 #$Person)
(#$loves ?PERSON1 ?PERSON2))),
```

in English, means

```
"Everybody loves somebody."
```

In Cyc, a collection means a group or class. Collections have instances. Each instance represents an individual. For examples,

```
(#$isa #$AbrahamLincoln, #$Person).
(#$isa #$BillGates, #$Person).
```

Abraham Lincoln and Bill Gates are individuals. Person is a collection. A collection could be an instance of another collection. For example,

```
(#$genls #$Dog, #$Mammal),
```

means "Collection Dog is an instance collection of collection Mammal".

In other word, Dog is a specialization of Mammal. It can be said that every individual is an instance of Thing, which is the most general collection in Cyc KB. Some individuals could be part of other individuals. For example, Microsoft is an individual. Joe works for Microsoft. Joe is part of Microsoft.

Constants are the "vocabulary words" of the Cyc KB, standing for something or a concept in the world that many people could know about. For example, #$isa, #$Person and #$BillGates are constants.

An assertion is the fundamental unit of knowledge in the Cyc KB. According to [41], every assertion consists of:

- an expression in CycL language that makes some declarative statement about the world.
- a truth value which indicates the assertion's degree of truth. There are five possible truth values, including *monotonically true*, *default true*, *unknown*, *default false* and *monotonically false.*
- A microtheory of which the assertion is part of. Section 3.2.1 gives a detailed explanation of microtheories.
- A direction which determines whether inferences involving the assertion are done at assert time or at ask time. There are three possible values for direction: *forward* (inferences done at assert time), *backward* (inferences done at ask time), and *code* (assertion not used in regular inference).
- A justification which is the argument or set of arguments supporting the assertion's having a particular truth value.

An assertion could be a rule or a Ground Atomic Formula (GAF). A rule is any CycL formula which begins with #$implies. A GAF is a CycL formula of the form, (predicate arg1 [arg2 ...argn]), where the arguments are not variables.

In Cyc, time is part of the upper ontology. It is a physical quantity. A temporal object such as an event, a process, or any physical object has a temporal extent. The time model is interval-based with support for points. Time Interval has dates, years, and so on, as its subcategories. An event is a set of assertions that describe a dynamic situation in which the state of the world changes. An event has non-empty space and time components. It may also have performer, beneficiaries, or victims. A script in CycL is a type of complex event with temporally-ordered sub-events. Applications can use script recognition - that allows them to identify a larger script from some stated events that are constituent parts of the script. Scripts can also be used for planning and for reading comprehension.

3.2.1 Microtheories

A microtheory (Mt) [41] is a bundle of assertions. The bundle of assertions may be grouped based on shared assumptions, common topic (music, football, etc), or source (Newsweek, People's Daily, etc). The assertions within a Mt must be mutually consistent. Assertions in different Mts may be inconsistent. For example,

MT1: Hu Jintao is the President of China

MT2: Hu Jintao is a high school student

Microtheories are a good way to cope with global inconsistence in the KB, providing a natural way to represent things like different points of views, or the change of scientific theories over time. Mts are one way of indexing all the assertions in Cyc KB.

There are two special Mts, one is #$BaseKB (always visible to all other Mts), the other one is #$EverythingPSC (all other Mts are visible to this Mt). #$EverythingPSC is a microtheory which has no logically consistent meaning but has a practical utility just because it is able to see the assertions in every microtheory.

The Cyc KB is the repository of Cyc's knowledge. It consists of constants and assertions involving those constants. It could be regarded as a sea of assertions, see figure 3.1. Form ontology point of view, the

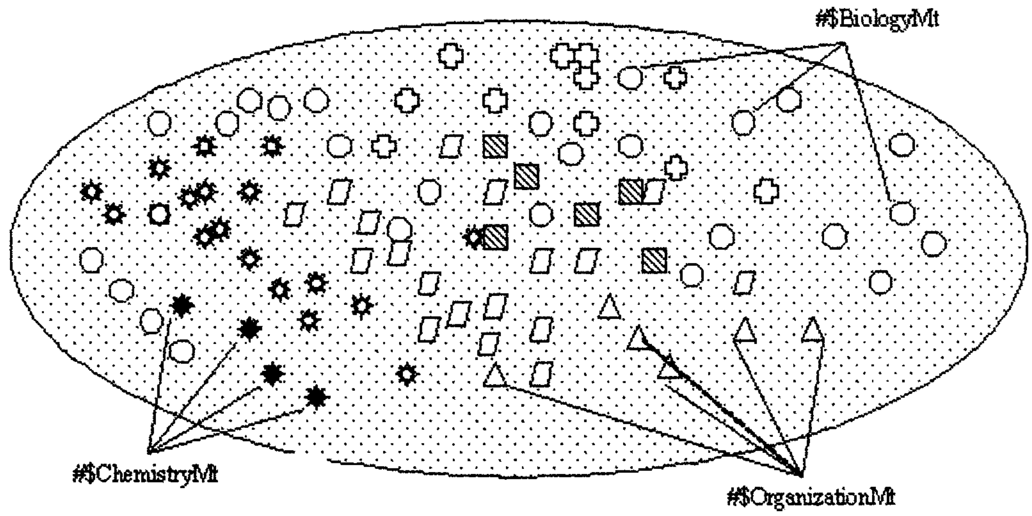

Figure 3.1: Cyc Knowledge Base as a Sea of Assertions

Cyc KB could also be thought of as made up of layers ordered by degree of generality.

3.2.2 Cyc Natural Language System (Cyc-NL)

Cyc-NL [41] is the natural language processing system associated with the Cyc KB. It could translate between natural language and CycL. Cyc-NL has three main components:

- a lexicon which is a generative morphology component generates part-of-speech assignments for words in a sentence.
- a syntactic parser which uses a grammar to generate all valid parses for the sentence.
- a semantic interpreter which produces pure CycL equivalent for the input sentence.

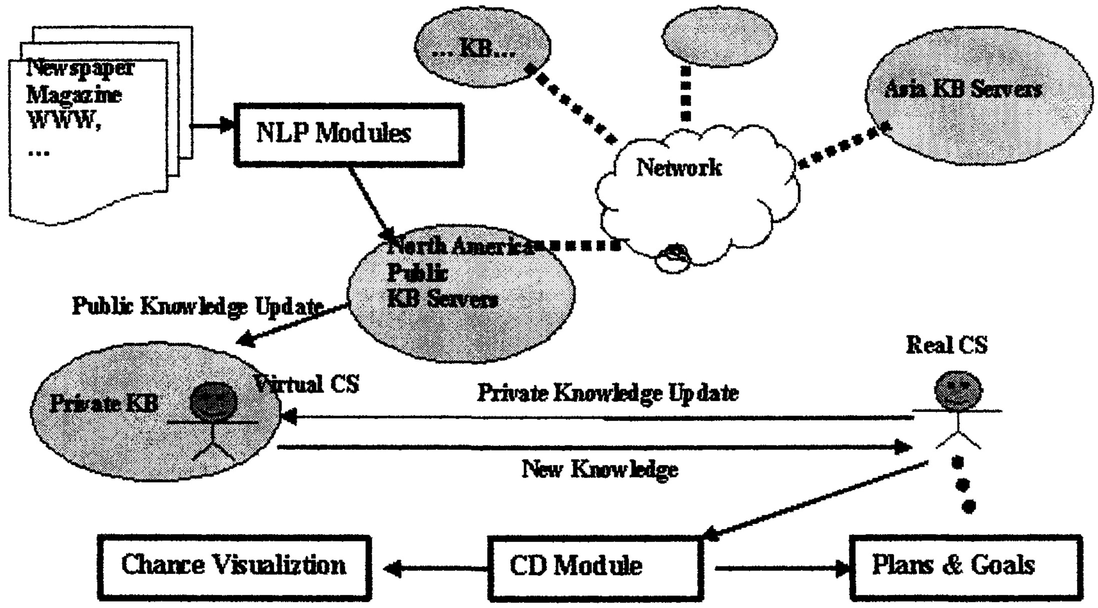

Figure 3.2: Chance Discovery System

3.3 Overview of Knowledge-based Chance Discovery System

Figure 3.2 shows the proposed framework for chance discovery. Natural Language Processing (NLP) modules analyze daily news and generate new knowledge which is represented in logic. The new knowledge is then integrated into public Cyc KB servers. The private Cyc KB server owned by the chance seeker will connect to public KB servers and update its knowledge. On the other hand, the chance seeker updates its private attributes in the private Cyc KB. The knowledge about the chance seeker can be regarded as a virtual chance seeker living in Cyc KB. A chance seeker sets up its goals or uses default goals in the Goals & Plans Module. New knowledge triggers the CD modules that measure the relevance of the new knowledge to the chance seeker. The new knowledge is considered to be a chance candidate if the relevance score is above a certain threshold. By trying to revise current plans using the new knowledge, the magnitude of this chance candidate can be measured using a utility evaluation process. When the magnitude of the utility is above a specified threshold, a chance is detected. Finally, the system visualizes the chances to chance seeker, and revises current plans for future chance detections.

3.4 The Relevance of New Knowledge

New knowledge is relevant to the chance seeker if it has an immediate impact on the seeker's attributes or on the achievability of the chance seeker's goals. For example, the new knowledge that shows that the chance seeker inherited a fortune is relevant as it changes the seeker's wealth attribute. The new information can affect the achievability of goals in three ways:

- making new goals achievable,
- making some previously achievable goals unattainable, or
- changing the cost or reward of achieving some goals.

A goal is considered achievable if the system finds a plan to the goal from the current state. To impact the achievability of a plan, the new knowledge could affect the causal support for actions in the plan or the likelihood of success.

Testing the relevance of new information to the chance seeker is desirable to filter out irrelevant information. Fully testing the relevance of new information with respect to its impact on the chance seeker's attributes and plans could be computationally expensive. Therefore, we gradually apply a series of relevance tests with increasing computational cost. These tests are:

- testing if the new information is subsumed by existing knowledge,
- testing for temporal relevance,
- testing for spatial relevance,
- testing for impact on the chance seeker's attributes, and
- testing for impact on the chance seeker's plans.

To verify that the new information is actually new, and is not subsumed by knowledge already in the KB, we test if it is entailed by existing knowledge. For example, if the KB contains assertions indicating that Paul Martin is the leader of the Liberal Party, that the Liberals won the largest number of seats in the parliament and that the leader of the party that wins the most seats becomes the Prime Minister. It becomes redundant to add an assertion indicating that Paul Martin became the Prime Minister. Similarly, if KB contains a generalization of the new information, this information will be redundant.

The relevance of information in a dynamic stochastic system degenerates gradually over time. The rate of degeneration of information relevance with respect to a rational decision maker depends on the probabilities of change as well as on the relative utilities [53]. Cyc supports a notion of possibility akin to probability. However, it is unlikely that the probabilistic knowledge in the KB will be specified fully to construct dynamic belief networks. Therefore, we rely on the intersection of the temporal extents associated with temporal object in the KB to verify the mutual relevance of temporal objects. Similarly, most spatial effects also weaken with distance. Therefore, it is fair to filter out new knowledge whose spatial or temporal effects lie outside the scope of interest.

New knowledge could be divided into rules and events (facts). We consider that the chance seeker relies on a rule if the chance seeker includes some actions that are causally supported by the consequences of the rule into its plan. The impact of the rule measures the role of the rule in reaching the goals. It could be regarded as the utility changes that are credited to the rule B. If S represents the state of chance seeker's attributes, then impact is given by:

$$Impact_B = V(S_B) - V(S) \tag{3.1}$$

To assess $V(S_B)$, we consider two cases: In one case, $V(S_B)$ may already be stated clearly in the rule. For example, the time saving from taking a newly built high speed train to a certain destination will be clearly stated in the news. On the other hand, if $V(S_B)$ is unclear, we can deduce a reasonable hypothesis by combining the new rule and existing rules in background KB. This hypothesis will not go beyond the known knowledge. For example, if there is an assertion in KB stating that all the people in the same country speak the same language, then communicating with all Brazilians will be the utility of learning Portuguese for a chance seeker who wants to travel to Brazil. Note that this utility could be inaccurate since it is based on a hypothesis. In general, $Impact_B$ may act as a greedy measure of progress towards the goals but does not guarantee reaching these goals. An exogenous rule may undermine actions on the other part of chance seeker.

When new knowledge is an event, to determine the value of an event, we have to take other factors into account. An event could be composed by a bundle of assertions describing its features, such as actions, locations, time, physical object involved, etc. The impact of an event with respect to a particular chance seeker is based on the following features:

- Importance of the entities involved in the event. To evaluate an event, we take the importance of those objects into account. For example, 'Microsoft' may be considered to be a more important company than other small companies. However, a small company currently working with Microsoft may be important.

- The relationship between involved objects and the chance seeker needs to be taken into account. For example, a company owned by family members may mean a lot to the chance seeker though it is a small company. On the other hand, the chance seeker may work for this small business. Generally, close relatives, friends, and acquaintances are more important than strangers.

According to the above:

$$Impact_{Event} = \sum_i V_E(Size(Object_i), Relations(Object_i, CS)) \quad (3.2)$$

Where V_E is a value function that takes into account the importance/size of object, the attributes involved and the relationships between objects and the chance seeker including spatio-temporal relationships. V_E tries to guess the potential change in the chance seeker's attributes.

A negative impact indicates that the new knowledge is a potential threat. In the case of irrelevant new knowledge, the impact will be inside the range of [negative threshold, positive threshold]. The new knowledge will be integrated into KB for future reference. However, the new knowledge will be considered as a chance candidate if the impact is outside the range.

3.5 The Magnitude of Chance Candidate

Here, B is the set of new knowledge that passes the relevance tests, the system will try to revise current plans (CP) of the chance seeker using B. Partial Order Planning (POP) and SATplan algorithm [46] can be used to generate new plans (NP_B) by taking B into account. In our system, we use SHOP [34] planner to generate the plans for the chance seeker. SHOP is a domain-independent automated-planning system. Chapter 4 will give a detail description of SHOP.

By adopting NP_B instead of CP, the chance seeker may be able to achieve a different set of goals, or save time and/or money while achieving the same goals. All these features can be reflected by a utility function mapping. The magnitude of B denoted by M_B is represented as the utility difference between NP_B and CP.

There could be a gap between the goals of NP_B and the goals of CS. As describing in section 3.1, a set of goals can be represented by a future status of attributes important to the chance seeker. If we use a utility function (V) to map those attributes into real values and add them together, we can represent a notion of preference. The change in the utilities could be represented as:

$$M_B = V_{NP_B} - V_{CP} \tag{3.3}$$

M_B represents the difference between new plans and current plans. If M_B in the range of [negative threshold, positive threshold], it means that NP_B and CP are roughly the same. The magnitude of B is low. Whether B is a chance or not, there are the following possible cases:

- **Short-term setback:** When B has negative effect on chance seeker's attribute and no threat to the current plans, B will be ignored.

- **Potential risk:** When B has negative effect on chance seeker, and threatens some of the current plans. However, repair plans can be found such that the new plans including the repair plans can achieve the same goal as before. This is considered a potential risk even though it is possible to repair the plans because if the chance seeker proceeds with the original plans the goals may not be reached.

- **Risk:** Repair plans cannot be found, NP_B achieve fewer goals than before. M_B is out of range. The system considers B to be is a risk.

- **Short-term prosperity:** When B has positive effect on chance seeker's attribute, and no effect on the current plans.

- **Exploitable efficiency:** NP_B can achieve the same goals as CP but in significantly shorter time or costs less. B is considered as a chance.

- **Improved reliability:** NP_B can achieve the same goals as before for approximately the same cost but offer an alternative for some plan elements.

- **Inefficient alternative:** Exploiting B, NP_B can achieve fewer goals than before or the same goals at a higher cost without threatning CP. B is ignored.

- **Opportunity:** NP_B can achieve more goals than before. M_B is significant and positive and B is considered a chance.

- **Short-term gain long-term risk:** When B has positive effect on chance seeker, threatens some of the current plans and the plans cannot be repaired.

- **Short-term loss long-term gain:** B results in an immediate loss but enables longer term plans.

Finally, if a chance is detected, NP_B will be set as CP.

3.6 Visualizing Chance

When a chance is detected, visualizing chances is important as the last step of chance discovery. Sometimes chance seekers may not understand why chances returned by chance discovery system are chances. Visualization of chances could emphasize on the explanation and help chance seeker realize chances.

A detail visualization explanation including display of the future status of attributes of chance seeker, display of chance seeker's current plans, etc, may be necessary. Kundu et al. [21] present a 3-D visualization technique for hierarchical task network plans. Such visualizations will be useful for the chance seeker to understand the interactions between various elements in the plan.

3.7 Evaluation & Discussion

The evaluation of chance discovery (CD) systems could be based on precision, efficiency and chance management. As discussed in Chapter 2, many previous CD approaches regard chances as unknown hypothesises, focusing on techniques to derive common chances, i.e. chances for all people. Our approach focuses on knowledge management, finding chances in known knowledge (news, WWW, etc) for a particular chance seeker by the support of a large and rich knowledge base. In the 2005 tsunami tragedy, scientists correctly detected the occurrence of the tsunami, but failed to warn the relevant people in South Asia in time to evacuate. Hence, chances are relative.

KeyGraph, as introduced in chapter 2, is a widely used technique in CD research. In section 2.4.2, we introduce a method proposed by Matsumura and Ohsawa to detect emerging topic (web page as chance) by applying KeyGraph on web pages. A "Human Genome project" example was presented. Its benefits include finding cures to conquer fatal illness. Two sets of web pages (CA and CB), each containing 500 web pages, were obtained by searching "human genome" in Google. CA was obtained on Nov 26, 2000. CB was on Mar 11, 2001. In the output of KeyGraph, Celera (www.celera.com), a growing Human Genome research website, was detected as a chance in CB because Celera cooccurred with the most important (foundation) websites in CB. The set of foundation websites of CA and CB, such as NCBI (the National Centre for Biotechnology Information), etc, is almost the same. The following events about Celera were reported in the meantime:

1. The Human Genome Project team and Celera announced the completion of the draft sequence of the human genome in June, 2000.

2. Craig Venter, President and Chief Scientific Officer of Celera and Francis Collins, Director of the Human Genome Project, met President Bill Clinton and British Prime Minister Tony Blair for the progress of the human genome analysis.

3. Papers about the completion were published in Nature and Science in Feb, 2001.

For a researcher in medicine whose goals include finding a cure for genetic diseases, our CD system would report a chance after evaluating events 1&2 and would propose new plans. The system may draw the researcher's attention to the draft sequence as early as on Jun 27, 2000 because Clinton and Blair are very important individuals. The degree of relevance will be high. The magnitude of "the draft sequence" will be high since it makes the researcher's unattainable goals achievable. Therefore, our approach could discover chances fast.

Chapter 4

Implementation of the Chance Discovery System

Having presented the theoretical aspects of our knowledge-based chance discovery system in the previous chapter, we now shift our attention to the implementation of the system.

4.1 Discussion on Implementation

To implement the chance discovery system, we need the support of the following essential elements:

- **Natural Language Processing Tool:** Translate knowledge between natural language text and logic representation.
- **Knowledge Entry Tool:** Automatically integrate knowledge represented in logic language into knowledge base.
- **Inference Engine:** Efficient inference engine in KB can reason logical consistence and answer inquiry by the point of view of logic.
- **Planner:** A planner generates a specific sequence of actions about

how to achieve goal state from initial state with respect to the knowledge defined in the knowledge base.

In the following sections, we will explore each element, present what we achieve and discuss the limitations.

4.1.1 Natural Language Processing Tools

> The goal of the Natural Language Processing (NLP) is to design and build software that will analyze, understand, and generate languages that humans use naturally, so that eventually you will be able to address your computer as though you were addressing another person.
> (Microsoft NLP Research Group)

Large-scale natural language processing (NLP) is a famously difficult task for many reasons. NLP research includes syntactic parsing, lexical representation, semantic interpretation, pragmatic processing, and discourse management. A truly functional natural-language dialogue system can be produced by incorporating the above-mentioned areas. Although important progresses have been made in these areas, NLP technology available today still has big difficulty in translating between nature language and logic language [41].

Cyc-NL [41], as introduced in section 3.2.2, is the natural language processing system associated with the Cyc knowledge base. Although benefited from a broad and deep repository of commonsense knowledge (Cyc KB), the Cyc-NL system in OpenCyc v0.7 can only 'speak English' but don't understand English well. Speaking English means that Cyc-NL can effectively translate CycL expressions into English.

Therefore, in the our system the translation is done manually.

4.1.2 Knowledge Entry in Cyc KB

There are several ways to integrate knowledge into Cyc KB. We can use web browser to access Cyc KB and add new knowledge by using:

- **Assert tool:** This tool can assert a formula written in CycL language into Cyc KB at one time. For example,

  ```
  (#$implies
  (#$isa ?SOMEONE #$Person)
  (#$isa ?SOMEONE #$Primate))
  ```

  is a well-formed formula and can be asserted into KB. A well-formed formula is a formula written with correct grammar of CycL.

- **Compose tool:** This tool allow users to enter new knowledge into Cyc KB using KE text format (KE stands for knowledge entry). KE text let user inputting massive knowledge at one time become possible. The following example show an example of KE text:

  ```
  Default Mt: BaseKB.
  Constant: WindsorLifeMt.
  isa: FictionalContext.
  genlMt:HumanActivitiesMt.

  Default Mt: WindsorLifeMt.

  Constant: Canada.
  isa: Country.

  Constant: Ontario.
  isa: State-Geopolitical.
  ```

```
Constant: WindsorArea.
isa: GeographicalRegion.

Constant: JohnLambton.
isa: ComputerProgrammer.

Constant: Java.
isa: ComputerLanguage.

Constant: Linux.
isa: ComputerProgram-CW.

Constant: Master-ComputerScience.
isa: AttributeValue.

F: (knowsAbout JohnLambton Java).
F: (knowsAbout JohnLambton Linux).
F: (residesInRegion JohnLambton WindsorArea).
F: (hasAttributes JohnLambton Master-ComputerScience).
```

This example builds a Microthieory 'WindsorLifeMt' enters the following knowledge into 'WindsorLifeMt':

> John Lambton lives in Windsor area, Ontario, Canada. He is a computer programmer, knowing about Java and Linux. John is holding Master degree in computer science.

Therefore, using KE text is good way to integrate new knowledge in our CD system.

4.1.3 Inference Engine in Cyc KB

Inference is the mechanism to conclude new facts from other existing facts and rules in the system [41]. The inference process is a deduction using facts and rules. Cyc uses two rules of inference in theorem proving, *modus ponens* and *modus tollens*. As we describe in chapter 3.2.1, Using microtheories (Mts) is a good way to incorporate local consistency while coping with global inconsistency. Inference is performed within Mts. Cyc Inference Engine can perform the following functions:

- **Answer Query:** For example, if we ask
  (#$isa #$JohnLambton ?X),
  Cyc KB will return the followings:

  ```
  Answer ?X

  *[Explain #55] ComputerProgrammer
  *[Explain #54] Individual
  *[Explain #53] Thing
  *[Explain #52] ComputerUser
  *[Explain #51] Agent-Generic
  ......
  *[Explain #32] OrganicStuff
  *[Explain #31] NaturalTangibleStuff
  *[Explain #30] EukaryoticOrganism
  *[Explain #29] Organism-Whole
  *[Explain #28] Heterotroph
  *[Explain #27] PerceptualAgent
  *[Explain #26] IndividualAgent
  *[Explain #25] Agent
  ......
  *[Explain #1] LegalAgent
  *[Explain #0] SocialBeing
  ```

  In previous example, we only asserted 'JohnLambton is computer programmer'. The inference engine can deduce the above knowledge by using existing rules. JohnLambton is an instance of the above collections.

- **Maintain Consistency:** If we try to assert a new assertion:

```
(isa JohnLambton Country) in WindsorLifeMt
```

  Beause John cannot be a person and country at the same time, Cyc KB will return the following:

```
Agenda halted due to:
 sbhl conflict:
(isa JohnLambton Country) :TRUE WindsorLifeMt

because: (isa JohnLambton ComputerProgrammer)
True-JustificationTruth
 (genls ComputerProgrammer Person) :TRUE
 (genls Person Animal) :TRUE
 (genls Animal AnimalBLO) :TRUE
 (genls AnimalBLO BiologicalLivingObject) :TRUE
 (genls InanimateThing-NonNatural InanimateThing) :TRUE
 (genls Organization InanimateThing-NonNatural) :TRUE
 (genls GeopoliticalEntity Organization) :TRUE
 (genls Country GeopoliticalEntity) :TRUE
```

  In this case, The new knowledge cannot be asserted into KB.

4.1.4 OpenCyc Java Application Programming Interface

OpenCyc Java Application Programming Interface (API) provide the middleware which allows Java applications to connect to Cyc KB and manipulate the knowledge in Cyc KB. Figure 4.1 shows the architecture of OpenCyc Java API. Class CfaslInputStream translate the input stream. All Java native types are translated to equivalent logical CycL language types. Class CfaslOutputStream does the opposite job. Figure 4.2 shows an example about how to access Cyc KB by using Java

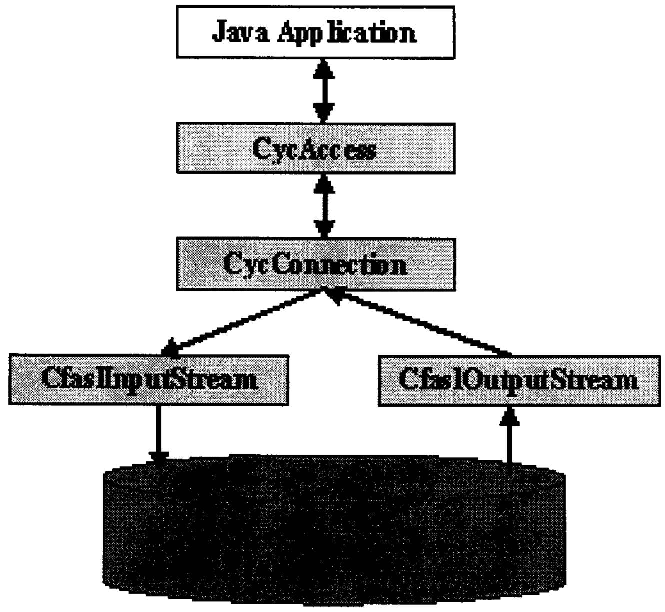

Figure 4.1: The OpenCyc Java API Architecture

application. In this example, the guid of constant is the unique identifier inside Cyc KB. We try to print out all the person instances in OpenCyc KB. We only got three instances: Guest, CycAdministrator and JohnLambton. Therefore, the knowledge in OpenCyc current version is very limited. All the examples about using OpenCyc Java API are available in Appendix A.

4.1.5 Planning

Since chance represents a new seed for a potentially significant future, we need an actual plan to show the potential way from current state to the significant future by taking advantage of the chance, i.e., We need to generate an actual plan to evaluate the magnitude of chance candidate. However, the OpenCyc planner does not work. Therefore, we implement our own planner outside the Cyc KB. This task was

```
package cycTestPackage;

import org.opencyc.api.CycAccess;
import org.opencyc.api.CycConnection;
import org.opencyc.cycobject.*;
import java.util.*;

public class Cyctest  {
 public Cyctest() {
 }

public static void main(String[] args) {
......
try {
        ......
      CycAccess connect=new CycAccess("localhost",3601,1,true,1);
      CycConstant person=(CycConstant)(connect.getConstantByName("Person"));

      CycList lis=connect.getAllIsa(person);
      CycList lis1=connect.getAllInstances(person);
      ......
      System.out.println(connect.baseKB.toXMLString());
      System.out.println(person.getGuid().toXMLString());
      for(Iterator it=lis1.iterator();it.hasNext();){
        System.out.println("Person isa "+((CycFort)(it.next())).toString());
      }
  ......
 } catch (Exception e)
     {
       System.out.println("error happened");
  }
 }
......
}
```

The following will be returned when we run the program:

```
<constant>
  <guid>bd588111-9c29-11b1-9dad-c379636f7270</guid>
  <name>BaseKB</name>
  <id>184</id>
</constant>

<guid>bd588092-9c29-11b1-9dad-c379636f7270</guid>

Person isa Guest
Person isa CycAdministrator
Person isa JohnLambton
```

Figure 4.2: A Sample Java Program

facilitated by the fact that the OpenCyc planner is a re-implementation of SHOP planner. In the following section, we will give a brief review about artificial intelligence planning.

Artificial Intelligence Planning Overview

In [46], the task of coming up with a sequence of actions that will achieve a goal is called planning. Planning includes *classical planning* and *nonclassical planning.* The *classical planning* environment are fully observable, deterministic, finite, static (change happens only when the agent acts), and discrete (in time, action, objects and effects). However, the *nonclassical planning* environments are partially observable or stochastic. In our chance discovery system, we use *classical planning* because we assume that our virtual society (KB) is fully observable and deterministic.

Hierarchical Task Network Planning

Complexity is a big issue when we do planning in such a vast knowledge base like Cyc. Hierarchical decomposition [46] is one of the most popular ways to deal with complexity. The key benefit of hierarchical decomposition is that, at each level of hierarchy, a computational task is reduced to a small number of activities at the next lower level, so that the computational cost of finding the plan for the problem is small. In non-hierarchical methods, it is completely impractical for large-planning problems because a task is reduced to a large number of individual actions. In Hierarchical Task Network (HTN) Planning, the initial plan is viewed as a very high level of description of the goal. Plans

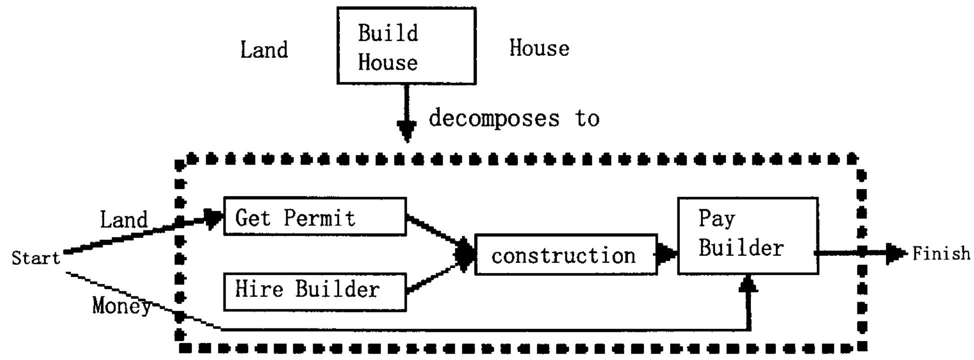

Figure 4.3: The One Possible Decomposition for the BuildHouse Action

are refined by applying action decompositions. Each action decomposition decomposes a high level action to a set of lower-level actions. This process embodies the knowledge about how to implement actions. It continues until only primitive actions remain in the plan. For example, figure 4.3 show the decomposition for the buildHouse action.

Simple Hierarchical Ordered Planner

SHOP (Simple Hierarchical Ordered Planner) [34] is a domain-independent automated-planning system. It is an ordered task decomposition planner which is a type of HTN planner and executes tasks in the same order as tasks appearing in plans. This reduces the complexity of reasoning by removing a great deal of uncertainty about the world and makes it easy to incorporate substantial expressive power into the planning algorithm.

SHOP is composed by the following elements [34]:

- *States* and *operators*: they represent knowledge about the status of domain and primitive actions agents can perform.
- *Task*: a task is an expression of any of the forms:

(s t_1 t_2 ... t_n)

(:task s t_1 t_2 ... t_n)

where s is a task symbol and the arguments t_1 t_2 ... t_n are terms. The task atom is

- *primitive*: Tasks we know how to execute directly. Normally, the task symbol is an operator name.
- *nonprimitive*: Tasks must be decomposed to subtasks using methods.

- *Methods*: A method is a list of the form

$$(\text{:method h } [n_1]\ C_1\ T_1\ [n_2]\ C_2\ T_2\ \ldots\ [n_k]\ C_k\ T_k)$$

where

- h is a task atom called the method's head.
- Each C_i is a logical precondition.
- Each T_i is a task list.
- Each n_i is the name for the succeeding C_i T_i pair.

A method indicates that the task specified in the method's head h can be performed by performing Ti when Ci is satisfied. The preconditions are considered in the given order, and a later precondition is considered only if all of the earlier preconditions are not satisfied. For example, C_2 is considered only if C_1 is not satisfied.

Figure 4.4 shows the basic elements of SHOP. A transport(package1,x,y) task can be decomposed to subtasks using two options: travel by train and by air.

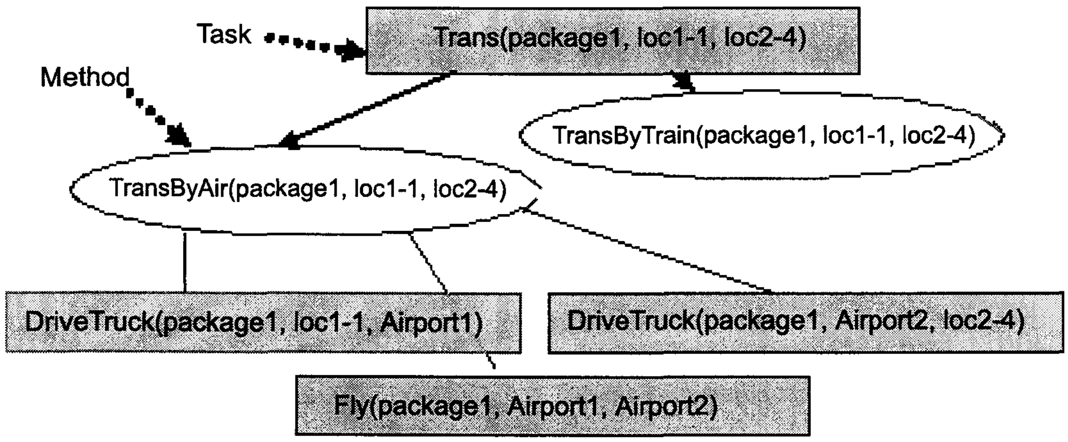

Figure 4.4: The Basic Elements of SHOP

To verify our theories, we have implemented a transportation domain and do planning by using SHOP planner, instead of using the planner of Cyc KB. Section 4.2 provides a detail case study about how chance is discovered in this transportation domain.

4.1.6 Limitations

From the above descriptions, we summarize that our theory is implementable in OpenCyc KB. However, the current available version of OpenCyc (v0.7) has many limitations. They are:

- It contains limited knowledge mostly describing about the upper level ontology.
- Although the OpenCyc API in OpenCyc (v0.7) presents a framework to manipulate Cyc KB server by using applications. We encountered a lot of bugs and empty method bodies (methods without the concrete code to fulfill the function that is supposed to have).

- The planner available in OpenCyc (v0.7) does not work.

Hence, the current version of Cyc KB does not satisfy all the requirements and the fact that it is a very large system makes attempting to fix its bugs beyond the scope of this thesis. However, a new version of Cyc KB, Cyc KB v1.0, is currently under development. Over fifty employees in Cycorp are developing this new version. We anticipate that it will be able to make use of this tool in the future.

4.2 Case Study

In this section, we demonstrate how example chances are detected in virtual transportation domain implemented by using SHOP. The Lisp code describing this domain along with its planning constraints is given in Appendix B.

4.2.1 Transportation Domain Description

This domain is based on logistic planning domain and talks about the transportation of package between cities, see Figure 4.5.

Due to the relative simplicity of the domain, chances we discover in this domain are not sophisticated and may seem trivial and simple. Another reason is that knowledge representation is not powerful enough in AI planners. SHOP planner does not have ontology hierarchy and does not support local consistency with global inconsistency. Therefore, 'chances' discovered here may not be what the human commonsense will consider a chance. However, they may seem reasonable by considering that these chances are discovered automatically.

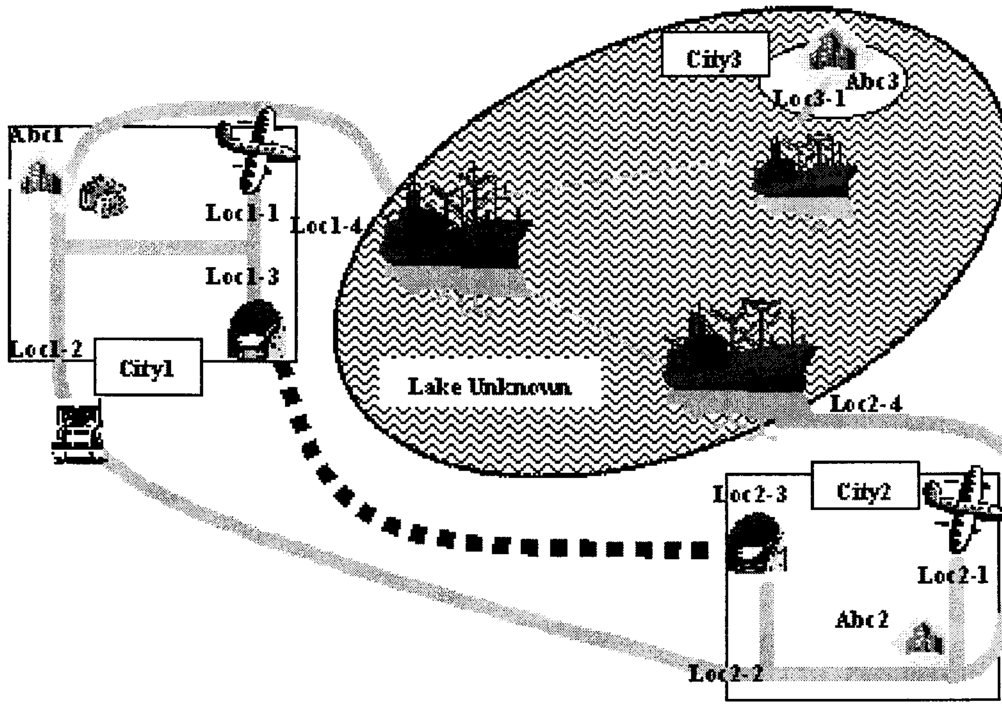

Figure 4.5: The Transportation Domain

In the future, with the support of powerful tools described in previous section, our chance discovery system should be able to discover valuable chance in the complex real world.

- **Domain Description:** Figure 4.5 shows ABC's transportation domain. ABC is a transportation company that has branches in cities and initially own a certain amount of money and a number of trucks.

  City1 and city2 are connected by a railway and a highway. Both cities have airports and docks. Daily flights and ferries travel between the two cities.

  City3, however, is located on an island and is only connected with other two cities only by ship.

ABC aim at achieving a daily transportation service, i.e., all packages need to be transported from origin to destination in one day. This goal is achieveable between city1 and city2 because ABC can use its own trucks and public services to transport packages. However, the goal is not achieveable for goods leaving and going to city3 due to the long travel time of the journey using a ship.

- **Goals:** The goal is to keep ABC in business and bring ABC profits as much as possible.
- **Chances:** By comparing a new plan and the current plan, we evaluate if new knowledge about change in the world is a chance or not. Chance examples could be:
  - A high speed train will serve between city1 and city2 soon.
  - A new cheap red eye flight will become available soon
  - Due to a serious flood, the highway between city1 and city2 will be closed for 2 months.

4.2.2 Chance Discovery Process

Step1: Translate new knowledge from natural language into logic language

Since Natural Language Processing tools available today still have big difficulties, the translation is done manually. For example, if the following news items become available:

1. the movie 'Crash' was awarded as Best Film at City1 Film Festival.
2. Due to a serious flood, the highway between city1 and city2 will be closed for 2 months.

3. a high speed cargo ship will serve between city1 and city3 soon.

4. Eric Lambton was erected to be the president of University of city2.

5. A new discovery shows that human being contacting virus X may die.

These news items represented in the logic language used in SHOP will be:

1. (MOVIE crash)
   (AWARD bestFilm)
   (FESTIVAL city1FilmFestival)
   (AWARD-TO bestFilm crash city1FilmFestival)

2. (COST-AT highway-cost 9999999 aFutureDate)

3. (COST-AT hs-ship 1200)
   (TIME-COST hs-ship 5)
   (SERVE-AT hs-ship city1 city3)

4. (PERSON EricLambton)
   (UNIVERSITY universityOfCity2)
   (PRESIDENT EricLambton universityOfCity2)

5. (VIRUS X)
   (EFFECT-ON X Person dead)
   (:operator (!CONTACT ?virus ?person)
   ((PERSON ?person)
   (STATUS ?person alive)
   (VIRUS ?virus)

(EFFECT-ON ?virus Person ?status)

((STATUS ?person alive))

((STATUS ?person ?status) 0)

Step2: The Relevance Test of New Knowledge

The relevance test in our system is to filter out the less relevant or irrelevant new knowledge with respect to chance seeker. As discussed in section 3.4, we need a deep and broad knowledge base to support this process. However, with the limited knowledge containing in OpenCyc (v0.7), this process can not be done. Therefore, the relevance test is also done manually. According to importance and relationship between these news items and ABC, the first, fourth and fifth news are considered as less relevant to the chance seeker(ABC). They may be filtered out in this step.

Step3: The Magnitude Test of Chance Candidates

News items that pass the relevance test are now considered as chance candidates and likely to become chances. In section 3.5, we have seen that there are ten possible cases when we use planning to evaluate chance candidates. This section evaluates a bundle of chance candidates which include the second and third news items in step1, describes how these candidates become real chances or ignored and sees how these candidates fit into one of the ten cases.

There is a plan called current plan (CP) which, according to current KB, is the best plan. CP achieves all of the chance seeker's goals or part of them. Initially, ABC only provides daily transportation service between city1 and city2, i.e., transports one package from abc1 to abc2 daily, see figure 4.5. Transporting by highway is the most economic way. The daily plan and quarter plan composed by daily plans are generated as follows:

- *Daily Plan:*

```
((!LOAD-TRUCK PACKAGE1 TRUCKA ABC1)
 (!DRIVE-TRUCK TRUCKA ABC1 LOC1-2)
 (!DRIVE-TRUCK-CITY TRUCKA LOC1-2 LOC2-2)
 (!DRIVE-TRUCK TRUCKA LOC2-2 ABC2)
 (!UNLOAD-TRUCK PACKAGE1 TRUCKA ABC2)
 (!MONEY-TIME-BALANCE 3000 850 2150 7))
```

  Where the last sentence means that in this plan, ABC earns 3000, spends 850, gains 2150 in profits; And it spends 7 hours in this journey.

- *Quarter Plan:*

```
((!WEEKDAY_EARNING_COST 3000 850)
 (!WEEKDAY_EARNING_COST 3000 850)
 (!WEEKDAY_EARNING_COST 3000 850)
 (!WEEKDAY_EARNING_COST 3000 850)
 (!WEEKDAY_EARNING_COST 3000 850)
 (!WEEKEND_EARNING_COST 3000 850)
 (!WEEKEND_EARNING_COST 3000 850)
 ......
 (!MONEY-BALANCE ABC 210650 15000 91))
```

  Where the last sentence means that after a 91 days' operation, ABC will have 210650 in assets when it originally has 15000 in assets. Here, one quarter of a year contains 13 weeks (91 days).

If no change occurs, ABC will follow the current plan to conduct its business. However, the environment is always changing and when a new chance candidate B becomes available, it may lead to one of the following cases:

- **Short-term setback**:
  - *Case 1*: Due to a terrorist threat, city1 airport will be closed for some days. Since we will use the highway for transportation, B will be ignored.
  - *Case 2*: Because of a serious flood, the highway between city1 and city2 will be close for 2 hours. In this case, ABC can just waits 2 hours and transport its packages by highway. B will be ignored.
- **Potential risk:** Because of a serious flood, the highway between city1 and city2 will be closed for 2 months. However, the railway is still on service. The system needs to change the current plan to use railway transportation. In these two months ABC's cost will rise a little bit. Although the Magnitude of B (M_B) is in the range, the system needs to inform ABC about the change of the plan. Part of the plan is as following:
  - *Daily Plan (revised)*:

```
((!DRIVE-TRUCK TRUCKA LOC1-3 ABC1)
 (!LOAD-TRUCK PACKAGE1 TRUCKA ABC1)
 (!DRIVE-TRUCK TRUCKA ABC1 LOC1-3)
 (!UNLOAD-TRUCK PACKAGE1 TRUCKA LOC1-3)
 (!RAILWAY-TRANSPORT PACKAGE1 LOC1-3 LOC2-3)
 (!DRIVE-TRUCK TRUCKB LOC2-1 LOC2-3)
 (!LOAD-TRUCK PACKAGE1 TRUCKB LOC2-3)
```

```
(!DRIVE-TRUCK TRUCKB LOC2-3 ABC2)
(!UNLOAD-TRUCK PACKAGE1 TRUCKB ABC2)
(!MONEY-TIME-BALANCE 3000 1200 1800 7.5))
```

– *Quarter Plan (revised):*

```
((!WEEKDAY_EARNING_COST 3000 850)
 (!WEEKDAY_EARNING_COST 3000 850)
 (!WEEKDAY_EARNING_COST 3000 850)
 (!WEEKDAY_EARNING_COST 3000 850)
 ......
 (!WEEKDAY_EARNING_COST 3000 1200)
 ......
 (!WEEKEND_EARNING_COST 3000 1200)
 (!WEEKEND_EARNING_COST 3000 850)
 ......
 (!MONEY-BALANCE ABC 189650 15000 91))
```

- **Risk:** City2 undergoes a serious strike. All public transits are out of service for 30 days. The transportation between city1 and city2 has to be stopped for 30 days. B is considered as a risk.

- **Short-term prosperity:** Due to highway condition improvements, the speed limit of the highway has been increased from 100km/h to 180km/h, which causes a considerable reduction in the travel time between cities. However, the time saved is not significant enough for a better plan, the cost is roughly the same and the daily work for ABC has not changed. So B has no effect on current plan. B is ignored.

- **Exploitable efficiency:** New cheap red-eye flights are opening. The cost will be greatly reduced if we take the red-eye air transport option. B is considered as a chance.

- **Improved reliability:** New cheap red-eye flights are opening but the cost of the new plan is roughly the same as the current plan. However, air transport option is more reliable and punctual when comparing to current plan using highway transport option. It could be regarded as a chance. However, currently B will be ignored.

- **Inefficient alternative:** A new cargo ship transit is opening but the ship travels at very slow speed. Although the cost is very low, it is not good for the transportation service ABC provide. Therefore, B will be ignored.

- **Opportunity:** Because of the long travel time, ABC currently has no transportation service to city3. However, A high speed cargo ship is going to serve the public. The travel time is shortened significantly. In this case, the ABC's transportation service to city3 becomes practical. Therefore, B is considered as a chance. We can have the following plans:

  – *Daily Plan (revised)*:

```
((!LOAD-TRUCK PACKAGE1 TRUCKA ABC1)
 (!DRIVE-TRUCK TRUCKA ABC1 LOC1-1)
 (!UNLOAD-TRUCK PACKAGE1 TRUCKA LOC1-1)
 (!RED-AIR-TRANSPORT PACKAGE1 LOC1-1 LOC2-1)
 (!DRIVE-TRUCK TRUCKB ABC2 LOC2-1)
 (!LOAD-TRUCK PACKAGE1 TRUCKB LOC2-1)
 (!DRIVE-TRUCK TRUCKB LOC2-1 ABC2)
 (!UNLOAD-TRUCK PACKAGE1 TRUCKB ABC2)
 (!DRIVE-TRUCK TRUCKA LOC1-1 ABC1)
 (!LOAD-TRUCK PACKAGE2 TRUCKA ABC1)
 (!DRIVE-TRUCK TRUCKA ABC1 LOC1-4)
 (!UNLOAD-TRUCK PACKAGE2 TRUCKA LOC1-4)
 (!HS-SHIP-TRANSPORT PACKAGE2 LOC1-4 LOC3-1)
 (!DRIVE-TRUCK TRUCKC ABC3 LOC3-1)
```

```
(!LOAD-TRUCK PACKAGE2 TRUCKC LOC3-1)
(!DRIVE-TRUCK TRUCKC LOC3-1 ABC3)
(!UNLOAD-TRUCK PACKAGE2 TRUCKC ABC3)
(!MONEY-BALANCE 5000 1670 3330 11))
```

– *Quarter Plan (revised):*

```
((!WEEKDAY_EARNING_COST 5000 1670)
 (!WEEKDAY_EARNING_COST 5000 1670)
 (!WEEKDAY_EARNING_COST 5000 1670)
 (!WEEKDAY_EARNING_COST 5000 1670)
 ......
 (!MONEY-BALANCE ABC 318030 15000 91))
```

- **Short-term gain long-term risk:** A high-speed cargo ship is going to serve between city1 and city2 soon. The cost and travel time are very attractive. However, the docks of city1 and city2 are far away from the city centers and the condition of the roads connecting them is very poor. In the long run, the maintenance cost of the trucks will be greatly raised. Therefore, it will not be a good deal to take this option. B will be ignored.

- **Short-term loss long-term gain:** When the same chance candidate as above comes, different situations make things different. The current Plan uses red-eye air transport option. This high-speed ship transport (hs-ship-transport) option cost more than red-eye transport option. Furthermore, the dock is far away from the city center. However, the docks and the city centers are connected by good condition expressways. Comparing to the poor local traffic condition, the maintenance cost of the trucks will be greatly reduced. Therefore, if we adopt hs-ship-transport option, we can get a better plan. B is regarded as a chance.

Chapter 5

Conclusion & Future Work

- **Conclusion:**

  In this thesis, we

  - summarized and discussed the achievement and limitations in previous approaches of chance discovery; proposed that chance is not necessary as unknown hypothesis, chance is typically associated with change and chance is relative to a person or an organization.
  - proposed a knowledge-based chance discovery system by combining the three necessary roles: communication, imagination and data mining. A knowledge base works as a virtual reality and simulates the development of real society by continuously updating its knowledge. The knowledge includes public knowledge and private knowledge about the chance seeker. This process can be regarded as a virtual chance seeker lives in this virtual society. The new knowledge comes from newspaper, magazine, and WWW, etc. The chance discovery system searches chances in KB for on behalf of the virtual chance

seekers. By assessing the relevance of new knowledge, the irrelevant knowledge to a chance seeker is ignored. Then chance in relevant knowledge is detected by considering its impact on the current plans and the possibility of new plans that are built based on the new knowledge. Finally, chance is visualized by displaying the future status of the chance seeker. Chance visualization helps the chance seeker to realize chance.

  - discussed and evaluated the implementation of the chance discovery system. Example chances are detected in a virtual transportation domain implemented by using SHOP planner.

- **Limitation:**
  - A full implementation of this knowledge-based chance discovery is currently unachieveable due to the absence of powerful tools described in Chapter 4.

- **Future Work:**
  - With the release of Cyc KB v1.0, we will implement the whole system in the future. We will build a virtual chance seeker in Cyc KB on behalf of a real business company and detect chances for the company in the real complex world over a long period of time.

Bibliography

[1] A. Abe. Applications of abduction. *Proc. of ECAI98 Workshop on Abduction and induction in AI*, pages 12–19, 1998.

[2] A. Abe. Abductive analogical reasoning. *Systems and Computers in Japan*, pages 11–19, 2000.

[3] A. Abe. The role of abduction in chance discovery. *Journal of New Generation Computing*, 21:61–71, 2002.

[4] A. Abe. Abduction and analogy in chance discovery. In Y. Ohsawa and P. McBurney, editors, *Chance Discovery*, pages 231–247. Springer-Verlag Berlin Heidelberg, 2003.

[5] R. Albert, H. Jeong, and A.L. Barabasi. The diameter of the world wide web. *Nature*, 401:130–131, 1999.

[6] W. R. Ashby. *An Introduction to Cybernetics.* Chapman & Hall, London, 1956.

[7] L. Bertalanffy. *General System Theory - A Critical Review.* New York, 1979.

[8] R. Brooks. A robust layered control system for a mobile robot. *IEEE Transactions on Robotics and Automation*, 2(1):14–23, 1986.

[9] A. Brown. Metacognition, executive control, self-regulation and other more mysterious mechanisms. In R. Kluwe and F. Weinert, editors, *Metacognition, Motivation and Human Performace*, pages 65–72. Lawrence Elbaum Associates, Hillsdale, 1987.

[10] E. Dietrich, A. B. Markman, C. H. Stilwell, and M. Winkley. The prepared mind: the role of representational change in chance discovery. In Y. Ohsawa and P. McBurney, editors, *Chance Discovery*, pages 208–229. Springer-Verlag Berlin Heidelberg, 2003.

[11] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. Knowledge discovery and data mining: Towards a unifying framework. *In Proceedings 2th International Conference on Knowledge Discovery and Data Mining*, pages 82–88, 1996.

[12] U. M. Fayyad, G. Piatestsky-Shapiro, and P. Smyth. *From Data Mining to Knowledge Discovery: An Overview.* AAAIMIT Press, 1996.

[13] FIPA. *Communicative Act Library Specification. Technical Report XC00031F, Foundation for Intelligent Physical Agents*, August 2001.

[14] J. Flavell. Metacognitive aspects of problem solving. *The Nature of Intelligence*, pages 231–235, 1976.

[15] J. Forester. *The Deliberative Practitioner: Encouraging Participatory Planning Processes.* MIT Press, Cambridge, MA, USA, 1999.

[16] J. Gavelek and T. Raphael. Metacognition, instruction, and the role of questioning activities. *Metacognition, Motivation and Human Performace*, 1985.

[17] D. Gentner. Analogical inference and analogical access. In A. Prieditis, editor, *Analogica*, pages 63–88. Pitman, London, UK, 1988.

[18] H. Kautz, B. Selman, and M. Shah. The hidden web. *AI magazine*, 18:27–36, 1997.

[19] S Kripke. Semantical analysis of modal logic 1. *Normal modal propositional calculii. Zeitschrift f.Math. Logik und Grunlagen d.Math.*, 9:67–96, 1963.

[20] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the web for emerging cyber-communities. *Proceedings of the 8th World Wide Web Conference*, 1999.

[21] K. Kundu, C. Sessions, M. desJardins, and P. Rheingans. Three-dimensional visualization of hierarchical task network plans. In *Proceedings of the Third International NASA Workshop on Planning and Scheduling for Space.* Houston, Texas, 2002.

[22] N. Matsumura. Topic diffusion in a community. In Y. Ohsawa and P. McBurney, editors, *Chance Discovery*, pages 84–96. Springer-Verlag Berlin Heidelberg, 2003.

[23] N. Matsumura, Y. Matsuo, Y. Ohsawa, and M. Ishizuka. Discovering emerging topics from www. *Journal of Contingencies and Crisis Management*, 10:73–81, 2002.

[24] N. Matsumura and Y. Ohsawa. Chance discoveries from the www. In Y. Ohsawa and P. McBurney, editors, *Chance Discovery*, pages 325–337. Springer-Verlag Berlin Heidelberg, 2003.

[25] N. Matsumura and Y. Ohsawa. Pai:automatic indexing for extracting asserted keywords from a document. *Journal of New Generation Computing*, 21:37–47, 2003.

[26] N. Matsumura, Y. Ohsawa, and M. Ishizuka. Future directions of communities on the web. *New Frontiers in Artificial Intelligence LNAI 2253*, pages 435–444, 2001.

[27] Y. Matsuo, Y. Ohsawa, and M. Ishizuka. A document as a small world. *New Frontiers in Artificial Intelligence LNAI 2253*, pages 444–449, 2001.

[28] R. McArthur and P. Bruza. Dimensional representations of knowledge in an online community. In Y. Ohsawa and P. McBurney, editors, *Chance Discovery*, pages 98–112. Springer-Verlag Berlin Heidelberg, 2003.

[29] R. McArthur and P. Bruza. Discovery of tacit knowledge and topical ebbs and flows within the utterances of an online community. In Y. Ohsawa and P. McBurney, editors, *Chance Discovery*, pages 115–131. Springer-Verlag Berlin Heidelberg, 2003.

[30] P. McBurney and S. Parsons. Chance discovery using dialectical argumentation. *New Frontiers in Artificial Intelligence LNAI 2253*, pages 414–424, 2001.

[31] P. McBurney and S. Parsons. Games that agents play: A formal framework for dialogues between autonomous agents. *Journal of Logic, Language and Information*, 11(3):315–334, 2002.

[32] P. McBurney and S. Parsons. Agent communications for chance discovery. In Y. Ohsawa and P. McBurney, editors, *Chance Discovery*, pages 133–146. Springer-Verlag Berlin Heidelberg, 2003.

[33] P. McBurney, S. Parsons, and M. Wooldridge. Desiderata for agent argumentation protocols. In C. Castelfranchi and W.L. Johnson, editors, *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AA-MAS 2002)*, pages 402–409. ACM Press, 2002.

[34] D. Nau, Y. Cao, A. Lotem, and H. Munoz-Avila. Shop: Simple hierarchical ordered planner. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 968–973, 1999.

[35] R. Oelmann. Metacognitive and computational aspects of chance discovery. *Journal of New Generation Computing*, 21:3–12, 2003.

[36] Y. Ohsawa. Keygraph as risk explorer in earthquake-sequence. *Journal of Contingencies and Crisis Management*, 10:119–128, 2002.

[37] Y. Ohsawa. Keygraph: Visualized structure among event clusters. In Y. Ohsawa and P. McBurney, editors, *Chance Discovery*, pages 262–275. Springer-Verlag Berlin Heidelberg, 2003.

[38] Y. Ohsawa. Modeling the process of chance discovery. In Y. Ohsawa and P. McBurney, editors, *Chance Discovery*, pages 2–15. Springer-Verlag Berlin Heidelberg, 2003.

[39] Y. Ohsawa, Nels E. Benson, and Masahiko Yachida. Keygraph: Automatic indexing by co-occurrence graph based on building construction metaphor. *Proceedings of the IEEE Forum on Reasearch and Technology Advances in Digital Libraries*, pages 12–18, 1998.

[40] Y. Ohsawa and P. McBurney. Preface. In Y. Ohsawa and P. McBurney, editors, *Chance Discovery.* Springer-Verlag Berlin Heidelberg, 2003.

[41] OpenCyc.org. Opencyc documentation. http: // www. openCyc. org /doc/, 2005.

[42] C. S. Peirce. *Philosophical Writings of Peirce*, chapter 11. NY:Dover, 1955.

[43] J. Perry. *Companion to the Philosophy of Language*, chapter Indexicals and Demonstratives. Oxford: Blackwell, 1997.

[44] H. Prendinger and M. Ishizuka. Methodological considerations on chance discovery. *New Frontiers in Artificial Intelligence LNAI 2253*, pages 425–435, 2001.

[45] S. Russell. *The Use of Knowledge in Analogy and Induction.* Pitman, London, UK, 1988.

[46] S. Russell and P. Norvig. *Artificial Intelligence A Modern Approach.* Prentice Hall, Second edition, 2003.

[47] G Schurz. Normic laws as system laws: Foundations of nonmonotonic reasoning. *Proceedings 4th Dutch-German Workshop on Nonmonotonic Reasoning Techniques and Their Applications*, pages 112–116, 1999.

[48] H. Shoji. Human-to-human communication for chance discovery in business. In Y. Ohsawa and P. McBurney, editors, *Chance Discovery*, pages 72–83. Springer-Verlag Berlin Heidelberg, 2003.

[49] H. Shoji and K. Hori. Chance discovery by creative communications observed in real shopping behaviors. *New Frontiers in Artificial Intelligence LNAI 2253*, pages 410–462, 2001.

[50] H. Shoji and K. Hori. Creative communication for chance discovery in shopping. *Journal of New Generation Computing*, 21:73–86, 2003.

[51] Y. Sumi and K. Mase. Enhancing daily conversations. In Y. Ohsawa and P. McBurney, editors, *Chance Discovery*, pages 304–323. Springer-Verlag Berlin Heidelberg, 2003.

[52] A.Y. Tawfik. Inductive reasoning and chance discovery. *Minds and Machines*, Volume 14(Issue 4):441 – 451, November 2004.

[53] A.Y. Tawfik and S. Khan. The degeneration of relevance in dynamic decision networks with sparse evidence. *Applied Intelligence*, page to appear, 2005.

[54] D.N. Walton and E.C.W. Krabbe. Commitment in dialogue: Basic concepts of interpersonal reasoning. *SUNY Series in Logic and Language*, pages XII+223, 1995.

[55] D. Watts. *Small worlds: the dynamics of networks between order and randomness.* Princeton University Press, 1999.

[56] D. Watts and S. Strogatz. Collective dynamics of small-world networks. *Nature*, pages 393–396, 1998.

[57] G.M. Weiss and H. Hirsh. Learning to predict rare events in event sequences. *Proc. of KDD-98*, pages 359–363, 1998.

[58] Z. Wu and A. Y. Tawfik. Towards knowledge-based chance discovery. In *Proceedings of the Seventh International Conference on Enterprise Information Systems (ICEIS)*, 2005.

Appendix A

1. An example to print out all the human instances in Cyc KB by using OpenCyc Java API.

```
package cycd;

/**
 * @author zhiwen wu
 *
 * This program will connect to OpenCyc server and print out all the instances of
*Person.
 */
import org.opencyc.api.CycAccess;
import org.opencyc.api.CycConnection;
import org.opencyc.cycobject.*;
import java.util.*;

package cycd;
import org.opencyc.api.CycAccess;
import org.opencyc.api.CycConnection;
import org.opencyc.cycobject.*;
import java.util.*;

public class Cyctest1  {
        public Cyctest1() {
        }

        public static void main(String[] args) {
                try {
                CycAccess cyc_a=new CycAccess("localhost",3601,1,ture,1);
                CycConstant cyc_c;

                CycConstant cyc_c=
               (CycConstant)(cyc_a.getConstantByName("Person"));
                CycList lisl=cyc_a.getAllInstances(cyc_c);
                 for(Iterator it=lis1.iterator();it.hasNext();){
                    System.out.println("Person isa "+((CycFort)(it.next())).toString());
                }

                } catch (Exception e) {
                    System.out.println("error happened");
                }
        }
}
```

2. An example tries to integrate knowledge from an xml file released by Cycorp using OpenCyc Java API.

```
package cycd;

/**
 * @author zhiwen wu
 *
```

```
 * The program try to integrate knowledge from an xml file released by Cycorp.
 * It works fine but very slow.
 */
import org.opencyc.api.*;
import java.io.*;
import java.lang.*;
import org.opencyc.cycobject.*;
import java.net.*;
import java.text.SimpleDateFormat;
import java.util.*;
import org.opencyc.xml.*;
import org.xml.sax.*;
import com.hp.hpl.jena.rdf.arp.*;
import com.hp.hpl.mesa.rdf.jena.common.*;
import com.hp.hpl.mesa.rdf.jena.model.*;
import org.opencyc.cycobject.*;
import ViolinStrings.Strings;

public class Cyctest2 extends ImportDaml {
	public Cyctest2() {
	}
	public Cyctest2(String input, String mt, CycAccess wu) {
		try {
			this.cycAccess = wu;
			importDaml(input, mt);

		} catch (Exception e) {
			System.out.println("input fail");
			e.printStackTrace();
			//exit(1);
		}
	}
	public void importDaml(
		String damlOntologyDefiningURLString,
		String importMtName)
		throws IOException, CycApiException {
		this.damlOntologyDefiningURLString = damlOntologyDefiningURLString;
		this.importMtName = importMtName;
		this.characterEncoding = null;
		if (verbosity > 0) {
			if (characterEncoding == null)
				System.out.println(
					"\nImporting "
						+ damlOntologyDefiningURLString
						+ "\ninto "
						+ importMtName);
			else
				System.out.println(
					"\nImporting "
						+ damlOntologyDefiningURLString
						+ " encoding "
						+ characterEncoding
						+ "\ninto "
						+ importMtName);
		}
		importMt = cycAccess.getKnownConstantByName(importMtName);
		damlOntologyDefiningURL =
			new CycNart(
				cycAccess.getKnownConstantByName("URLFn"),
				damlOntologyDefiningURLString);
		System.out.println(
			"Defining URL " + damlOntologyDefiningURL.cyclify());
		CycList gaf = new CycList();
		gaf.add(cycAccess.getKnownConstantByName("xmlNameSpace"));
		String nickname = "wuzhiwen";
		//	(String) ontologyNicknames.get(damlOntologyDefiningURLString);
		if (nickname == null)
```

```
                throw new RuntimeException(
                        "Nickname not found for " + damlOntologyDefiningURLString);
        gaf.add(nickname);
        gaf.add(damlOntologyDefiningURL);
        cycAccess.assertGaf(gaf, importMt);
        System.out.println("\nStatements\n");
        //cycAccess.traceOn();
        InputStreamReader in;
        URL url;
        try {
                File ff = new File(damlOntologyDefiningURLString);
                if (characterEncoding == null)
                        in = new InputStreamReader(new FileInputStream(ff));
                else
                        in =
                                new InputStreamReader(
                                        new FileInputStream(ff),
                                        characterEncoding);
                url = ff.toURL();
                System.out.println("url="+url.toString());
        } catch (Exception ignore) {
                try {
                        url = new URL(damlOntologyDefiningURLString);
                        if (characterEncoding == null)
                                in = new InputStreamReader(url.openStream());
                        else
                                in =
                                        new InputStreamReader(
                                                url.openStream(),
                                                characterEncoding);
                } catch (Exception e) {
                        System.err.println(
                                "ARP: Failed to open: " + damlOntologyDefiningURLString);
                        System.err.println(
                                "   " + ParseException.formatMessage(ignore));
                        System.err.println("   " + ParseException.formatMessage(e));
                        return;
                }
        }
        try {
                System.out.println(url.toExternalForm()+","+in.toString());
                arp.load(in, url.toExternalForm());
        } catch (IOException e) {
                System.err.println(
                        "Error: "
                                + damlOntologyDefiningURLString
                                + ": "
                                + ParseException.formatMessage(e));
        } catch (SAXException sax) {
                System.err.println(
                        "Error: "
                                + damlOntologyDefiningURLString
                                + ": "
                                + ParseException.formatMessage(sax));
        }
        catch(Exception e){
                System.out.println("something happened");
        }
        if (verbosity > 0)
                System.out.println(
                        "\nDone importing " + damlOntologyDefiningURLString + "\n");
}
public static void main(String[] args) {
        try {
                CycAccess wu=new CycAccess("localhost",3601,1,true,1);

                /* a test to see if connect is good */
```

```
                        wu.makeCycConstant("ABC");
                        wu.assertComment("ABC", "ABC is a good person", "BaseKB");

                        Cyctest2 know = new Cyctest2("H:/download/cyc1.xml", "WindsorLifeMt", wu);

                } catch (Exception e) {
                        System.out.println("error happened");
                }
        }
}
```

The following shows a part of the xml file:


```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns="http://www.cyc.com/2004/06/04/cyc#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <owl:Ontology rdf:about="">
    <owl:versionInfo>$Id: ExportOwl.java,v 1.11 2004/06/15 20:27:31
      reed Exp $</owl:versionInfo>
    <rdfs:comment>ResearchCyc Ontology  OpenCyc License Information
      The contents of this file constitute portions of The OpenCyc
      Knowledge Base. The OpenCyc Knowledge Base is protected
      under the following license and copyrights. This license and
      copyright information must be included with any copies or
      derivative works.  Copyright Information OpenCyc Knowledge
      Base Copyright 2001-2004 Cycorp, Inc., Austin, TX, USA. All
      rights reserved. OpenCyc Knowledge Server Copyright
      2001-2004 Cycorp, Inc., Austin, TX, USA. All rights
      reserved. Other copyrights may be found in various files.
      The OpenCyc Knowledge Base The OpenCyc Knowledge Base
      consists of code, written in the declarative language CycL,
      that represents or supports the representation of facts and
      rules pertaining to consensus reality. OpenCyc is licensed
      using the GNU Lesser General Public License, whose text can
      also be found on this volume. The OpenCyc CycL code base is
      the &quot;library&quot; referred to in the LGPL license. The
      terms of this license equally apply to renamings and other
      logically equivalent reformulations of the Knowledge Base
      (or portions thereof) in any natural or formal language.
      See http://www.opencyc.org for more information. </rdfs:comment>
  </owl:Ontology></rdf:RDF>
```

Appendix B

1. The Transportation Domain Description:

;;; Chance Discovery Transportation Domain –domain description, implemented by ZhiWen Wu, 2005

```
(defdomain Transportation
 (
          ;; basic operators
          (:operator (!load-truck ?obj ?truck ?loc)
                 ((total-cost abc ?cost)(assign ?c1 (call + ?cost 5)))
                 ((obj-at ?obj ?loc)(total-cost abc ?cost))
                 ((in-truck ?obj ?truck)(total-cost abc ?c1)) 5)

          (:operator (!unload-truck ?obj ?truck ?loc)
                 ((total-cost abc ?cost)(assign ?c1 (call + ?cost 5)))
                 ((in-truck ?obj ?truck)(total-cost abc ?cost))
                 ((obj-at ?obj ?loc)(total-cost abc ?c1)) 5)

          (:operator (!drive-truck ?truck ?loc-from ?loc-to)
                 ((cost-local ?loc-from ?loc-to ?cost)(total-cost abc ?tc)(assign ?c1 (call + ?tc ?cost)))
                 ((truck-at ?truck ?loc-from)(total-cost abc ?tc)(in-city ?loc-
        from ?city)(truck ?truck ?city))
                 ((truck-at ?truck ?loc-to)(total-cost abc ?c1))
                        ?cost)

          (:operator (!drive-truck-city ?truck ?loc-from ?loc-to)
                 ((TRUCK ?truck ?city-from)(IN-CITY ?loc-to ?city-to)
                  (cost-at highway-cost ?cost)(total-cost abc ?tc)(assign ?c1 (call + ?tc ?cost)))
                 ((truck-at ?truck ?loc-from)(total-cost abc ?tc)(TRUCK ?truck ?city-from))
                 ((TRUCK ?truck ?city-to)(truck-at ?truck ?loc-to)(total-cost abc ?c1))
                        ?cost)

          (:operator (!air-transport ?obj ?airport-from ?airport-to)
                 ((cost-at air-cost ?cost)(total-cost abc ?tc)(assign ?c1 (call + ?tc ?cost)))
                 ((obj-at ?obj ?airport-from)(total-cost abc ?tc))
                 ((obj-at ?obj ?airport-to)(total-cost abc ?c1))
                     ?cost )

          ;; red eye flight
          (:operator (!red-air-transport ?obj ?airport-from ?airport-to)
                 ((cost-at red-air-cost ?cost)(total-cost abc ?tc)(assign ?c1 (call + ?tc ?cost)))
                 ((obj-at ?obj ?airport-from)(total-cost abc ?tc))
                 ((obj-at ?obj ?airport-to)(total-cost abc ?c1))
                     ?cost )

          (:operator (!railway-transport ?obj ?station-from ?station-to)
                 ((cost-at railway-cost ?cost)(total-cost abc ?tc)(assign ?c1 (call + ?tc ?cost)))
                 ((obj-at ?obj ?station-from)(total-cost abc ?tc))
                     ((obj-at ?obj ?station-to)(total-cost abc ?c1))
                     ?cost)

          (:operator (!ship-transport ?obj ?dock-from ?dock-to)
                 ((cost-at ship-cost ?cost)(total-cost abc ?tc)(assign ?c1 (call + ?tc ?cost)))
                 ((obj-at ?obj ?dock-from)(total-cost abc ?tc))
                     ((obj-at ?obj ?dock-to)(total-cost abc ?c1))
                     ?cost)
```

```
 (:operator (!hs-ship-transport ?obj ?dock-from ?dock-to)
        ((cost-at hs-ship-cost ?cost)(total-cost abc ?tc)(assign ?c1 (call + ?tc ?cost)))
        ((obj-at ?obj ?dock-from)(total-cost abc ?tc))
            ((obj-at ?obj ?dock-to)(total-cost abc ?c1))
            ?cost)

 (:operator (!money-balance ?earning ?cost ?pure)
        ((total-cost abc ?cost)(assign ?pure (call - ?earning ?cost)))
            ()
            () 0)

;;;----------------------------------------------------------------------------
 (:operator (!add-protection ?g)
        ()
        ((:protection ?g))
        0)

 (:operator (!delete-protection ?g)
        ((:protection ?g))
        ()
        0)

 ;;;----------------------------------------------------------------------------

 ;; actual AI planning methods

 ;; take the airline
 (:method (obj-at ?obj ?loc-goal)
       same-city-deliver
       ((in-city ?loc-goal ?city-goal)
        (obj-at ?obj ?loc-now)
        (in-city ?loc-now ?city-goal)
        (truck ?truck ?city-goal))
           ((:task in-city-delivery ?truck ?obj ?loc-now ?loc-goal))

       different-city-deliver
           ((in-city ?loc-goal ?city-goal)
            (obj-at ?obj ?loc-now)
            (in-city ?loc-now ?city-now)
            (different ?city-goal ?city-now)
            (earn-money ?obj ?earning)
            (truck ?truck-now ?city-now)
            (truck ?truck-goal ?city-goal)
            (airport ?airport-now) (in-city ?airport-now ?city-now)(airport-open ?airport-now)
           (airport ?airport-goal) (in-city ?airport-goal ?city-goal)(airport-open ?airport-goal))
           (:ordered
                 (:task in-city-delivery ?truck-now ?obj ?loc-now ?airport-now)
                 (:task air-transport-obj ?obj ?airport-now ?airport-goal)
                 (:task in-city-delivery ?truck-goal ?obj ?airport-goal ?loc-goal)
                     (:task !money-balance ?earning ?cost ?pure) ))

;; drive truck cross town
(:method (obj-at ?obj ?loc-goal)
            drive-truck-cross-town
       ((in-city ?loc-goal ?city-goal)
        (obj-at ?obj ?loc-now)
        (in-city ?loc-now ?city-now)
        (different ?city-goal ?city-now)
        (earn-money ?obj ?earning)
```

```
            (truck ?truck ?city-now)
                (highway-entrance ?highway-from)(in-city ?highway-from ?city-now)
                (highway-entrance ?highway-to)(in-city ?highway-to ?city-goal) )

            (:ordered
                 (:task :immediate !load-truck ?obj ?truck ?loc-now)
                 (:task truck-at ?truck ?highway-from)
                 (:task !drive-truck-city ?truck ?highway-from ?highway-to)
                 (:task truck-at ?truck ?loc-goal)
                 (:task :immediate !unload-truck ?obj ?truck ?loc-goal)
                 (:task !money-balance ?earning ?cost ?pure)))

;;; go by train
 (:method (obj-at ?obj ?loc-goal)
         transport-by-train
        ((in-city ?loc-goal ?city-goal)
         (obj-at ?obj ?loc-now)
         (in-city ?loc-now ?city-now)
         (different ?city-goal ?city-now)
         (earn-money ?obj ?earning)
         (truck ?truck-now ?city-now)
             (truck ?truck-goal ?city-goal)
             (railway-station ?station-from)(in-city ?station-from ?city-now)
             (railway-station ?station-to)(in-city ?station-to ?city-goal))

        (:ordered
             (:task in-city-delivery ?truck-now ?obj ?loc-now ?station-from)
             (:task !railway-transport ?obj ?station-from ?station-to)
             (:task in-city-delivery ?truck-goal ?obj ?station-to ?loc-goal)
             (:task !money-balance ?earning ?cost ?pure)))

 ;;; transport by ship
 (:method (obj-at ?obj ?loc-goal)
         transport-by-ship-city12
        ((in-city ?loc-goal ?city-goal)
         (obj-at ?obj ?loc-now)
         (in-city ?loc-now ?city-now)
         (different ?city-goal ?city-now)
             (earn-money ?obj ?earning)
             (truck ?truck-now ?city-now)
             (truck ?truck-goal ?city-goal)
             (dock ?dock-from)(in-city ?dock-from ?city-now)
             (dock ?dock-to)(in-city ?dock-to ?city-goal)
             (different ?city-goal city3) )

        (:ordered
             (:task in-city-delivery ?truck-now ?obj ?loc-now ?dock-from)
             (:task !ship-transport ?obj ?dock-from ?dock-to)
             (:task in-city-delivery ?truck-goal ?obj ?dock-to ?loc-goal)
                 (:task !money-balance ?earning ?cost ?pure)))

 ;;; transport by high speed ship, city1 -city3
 (:method (obj-at ?obj ?loc-goal)
         transport-by-ship-city13
        ((in-city ?loc-goal ?city-goal)
         (obj-at ?obj ?loc-now)
         (in-city ?loc-now ?city-now)
         (different ?city-goal ?city-now)
             (earn-money ?obj ?earning)
             (truck ?truck-now ?city-now)
```

```
            (truck ?truck-goal ?city-goal)
            (dock ?dock-from)(in-city ?dock-from ?city-now)
            (dock ?dock-to)(in-city ?dock-to ?city-goal)
            (different ?city-goal city2) )

        (:ordered
             (:task in-city-delivery ?truck-now ?obj ?loc-now ?dock-from)
             (:task !hs-ship-transport ?obj ?dock-from ?dock-to)
             (:task in-city-delivery ?truck-goal ?obj ?dock-to ?loc-goal)
             (:task !money-balance ?earning ?cost ?pure)))

;;;---------------------------support functions---------------------------------
;;;----------------------------------------------------------------------------------

(:method (in-city-delivery ?truck ?obj ?loc-from ?loc-to)
        package-already-there
         ((same ?loc-from ?loc-to))
         ()

        truck-across-town
         ((in-city ?loc-from ?city)
          (truck ?truck ?city))
          (:ordered (:task truck-at ?truck ?loc-from)
                (:task :immediate !load-truck ?obj ?truck ?loc-from)
                (:task truck-at ?truck ?loc-to)
                (:task :immediate !unload-truck ?obj ?truck ?loc-to) ))

;;;------------------------------------------------------------------------------------
(:method (truck-at ?truck ?loc-to)

        truck-in-right-location
        ((truck-at ?truck ?loc-from)
         (same ?loc-from ?loc-to))
         ()

        truck-not-in-right-location
        ((truck-at ?truck ?loc-from)
         (in-city ?loc-from ?city)
         (in-city ?loc-to ?city)
         (different ?loc-from ?loc-to) )
         ((:task :immediate !drive-truck ?truck ?loc-from ?loc-to)))

;;;------------------------------------------------------------------------------------
(:method (air-transport-obj ?obj ?airport-from ?airport-to)
        take-off-and-fly-there
        ((obj-at ?obj ?airport-from)(airport ?airport-from)(airport ?airport-to))
         (:ordered (:task !air-transport ?obj ?airport-from ?airport-to)))

(:method (air-transport-obj ?obj ?airport-from ?airport-to)
        red-eye-air-line
        ((obj-at ?obj ?airport-from)(airport ?airport-from)(airport ?airport-to))
         (:ordered (:task !red-air-transport ?obj ?airport-from ?airport-to)))

;;;------------------------------------------------------------------------------------
;; state axioms

(:- (same ?x ?x) nil)
(:- (different ?x ?y) ((not (same ?x ?y))))

(:- (cost-local ?loc1 ?loc2 ?number)
```

```
        ((cost-c ?loc1 ?loc2 ?number))
        ((cost-c ?loc2 ?loc1 ?number)))

))
```

2. The Transportation Problem Description:

```
;;; Chance Discovery Transportation domain –problem description, implemented by ZhiWen Wu, 2005

(defproblem chanceDiscovery Transportation
 (
        ;; ----------description about city 1----------
        (CITY city1)
        (LOCATION abc1)
        (IN-CITY abc1 city1)
        (AIRPORT LOC1-1)
        (AIRPORT-OPEN LOC1-1)
        (TRUCK truckA city1)
        (TRUCK-AT truckA abc1)
        (LOCATION LOC1-1)
        (IN-CITY LOC1-1 city1)
        (LOCATION LOC1-2)
        (HIGHWAY-ENTRANCE LOC1-2)
        (IN-CITY LOC1-2 city1)
        (RAILWAY-STATION LOC1-3)
        (LOCATION LOC1-3)
        (IN-CITY LOC1-3 city1)
        (DOCK LOC1-4)
        (LOCATION LOC1-4)
        (IN-CITY LOC1-4 city1)

        ;;;-cost description-
        (cost-at air-cost 2000)
        (cost-at railway-cost 1000)
        (cost-at ship-cost 1100)
        (cost-at hs-ship-cost 1200)
        (cost-at highway-cost 800)
        (cost-at red-air-cost 700)

        (time-cost air 1.5)
        (time-cost railway 5)
        (time-cost ship 8)
        (time-cost hs-ship 5)
        (time-cost highway 6)
        (time-cost red-air 1.5)


        ;;--local cost based on distance and road condition-
        (cost-c abc1 loc1-1 60)
        (cost-c abc1 loc1-2 20)
        (cost-c abc1 loc1-3 60)
        (cost-c abc1 loc1-4 50)
        (cost-c loc1-1 loc1-2 60)
        (cost-c loc1-1 loc1-3 20)
        (cost-c loc1-1 loc1-4 110)
        (cost-c loc1-2 loc1-3 60)
        (cost-c loc1-2 loc1-4 70)
        (cost-c loc1-3 loc1-4 110)
```

```
;;-----------------------------------------
;; -------description about city 2---------
;;-----------------------------------------
(CITY city2)
(LOCATION abc2)
(IN-CITY abc2 city2)
(AIRPORT LOC2-1)
(AIRPORT-OPEN LOC2-1)
(TRUCK truckB city2)
(LOCATION LOC2-1)
(TRUCK-AT truckB abc2)
(IN-CITY LOC2-1 city2)
(LOCATION LOC2-2)
(HIGHWAY-ENTRANCE LOC2-2)
(IN-CITY LOC2-2 city2)
(RAILWAY-STATION LOC2-3)
(LOCATION LOC2-3)
(IN-CITY LOC2-3 city2)
(DOCK LOC2-4)
(LOCATION LOC2-4)
(IN-CITY LOC2-4 city2)

;;--local cost based on distance and road condition-
(cost-c abc2 loc2-1 60)
(cost-c abc2 loc2-2 20)
(cost-c abc2 loc2-3 60)
(cost-c abc2 loc2-4 50)
(cost-c loc2-1 loc2-2 60)
(cost-c loc2-1 loc2-3 20)
(cost-c loc2-1 loc2-4 110)
(cost-c loc2-2 loc2-3 60)
(cost-c loc2-2 loc2-4 70)
(cost-c loc2-3 loc2-4 110)

;;---------------------------------------
;;Description about city 3
;;---------------------------------------
(CITY city3)
(LOCATION abc3)
(IN-CITY abc3 city3)
(LOCATION loc3-1)
(DOCK loc3-1)
(IN-CITY loc3-1 city3)
(TRUCK truckC city3)
(TRUCK-AT truckC abc3)

(cost-c abc3 loc3-1 20)
(cost-at hs-ship-cost 600)

(earn-money package1 3000)
(earn-money package2 2000)

;; ---- money source----
(total-cost abc 0)
(obj-at package1 abc1)
(obj-at package2 abc1)

)
```

```
        ( :ordered
        ;; -----goal---------
        (:task obj-at package1 abc2)
        (:task obj-at package2 abc3)

))

(find-plans 'chanceDiscovery :which :all  :verbose 3)
;; :shallowest
```

3. Long term planning in Transportation Domain

```
;; Chance Discovery Transportation Domain –long term planning domain and
;; problem description, implemented by Zhiwen Wu, 2005.
;;; A quarter (1/4 year) plan is composed by Daily plans.

;;(:method (weekdays_transport ?num)
;;
;;      ((call < ?num 5))
;;        (:ordered
;;          (:task !weekday_earning_cost)
;;          (:task weekdays_transport (call + 1 ?num)))
;;
;;        Friday
;;        ((call = ?num 5))
;;      (:ordered
;;          (:task !weekday_earning_cost)))
;;
(defdomain yeartransport
  (
   (:operator (!weekday_earning_cost ?earn ?cost)
        ((cost-day-is ?cost)(earning-is ?earn)(has-money ABC ?asset)
            (assign ?a1 (call + ?asset (call - ?earn ?cost))))
        ((has-money ABC ?asset))
        ((has-money ABC ?a1)) 0)


  (:operator (!weekend_earning_cost ?earn ?cost)
        ((cost-day-is ?cost)(earning-is ?earn)(has-money ABC ?asset)
        (assign ?a2  (call + ?asset (call - ?earn ?cost))))
        ((has-money ABC ?asset))
        ((has-money ABC ?a2)) 0)

  (:operator (!own-money abc ?num)
       ()  () () 0)


   (:method (year_transport)
       weeks_transprot
       ()
       ((:task one_quarter_work 1)
          ;; (:task one_quarter_work 1)
           ;;(:task one_quarter_work 1)
          ;; (:task one_quarter_work 1)
          ))

   (:method (one_quarter_work ?num)
            final_week
           ((call = ?num 13))
```

```
          (:ordered
           (:task weekdays_transport 1)
           (:task weekends_transport))

       weeks_transprot
       ((call < ?num 13))
       (:ordered
           (:task weekdays_transport 1)
           (:task weekends_transport)
           (:task one_quarter_work (call + 1 ?num))
           ))

  (:method (weekdays_transport ?num)
     ()
        (:ordered
          (:task !weekday_earning_cost ?earn ?cost)
          (:task !weekday_earning_cost ?earn ?cost)
          (:task !weekday_earning_cost ?earn ?cost)
          (:task !weekday_earning_cost ?earn ?cost)
          (:task !weekday_earning_cost ?earn ?cost)))


  (:method (weekends_transport)
     ()
        (:ordered
          (:task !weekend_earning_cost ?earn ?cost)
          (:task !weekend_earning_cost ?earn ?cost)))

  (:method (print-current-state) ((eval (print-current-state))) ())
  (:method (print-current-tasks) ((eval (print-current-tasks))) ())
  (:method (print-current-plan) ((eval (print-current-plan))) ())
  (:method (print-asset) ((has-money abc ?asset)) (:ordered (:task !own-money abc ?asset)))
 ))

(defproblem year yeartransport
 (
 (cost-day-is 1670)
 (earning-is 5000)
 (has-money abc 15000)
 )

 (
 (year_transport)
 ;;(print-current-state)
 ;;(print-current-tasks)
 ;;(print-current-plan)
 (print-asset)
))

(find-plans 'year :which :first :verbose 2)
```

Vita Auctoris

Zhiwen Wu was born in 1973 in YunFu town, a lonely but beautiful mountainous region in South China. He spent his childhood in the mountains where his parents worked as teachers. At the age of twelve, he and his family moved to Jiangmen city which is located in Pearl River Delta, one of the most industrious regions in China. Zhiwen spent his high school time there. He graduated from the NO.1 middle school of Jiangmen city in 1992 and then went to Nanjing University for his undergraduate degree. Nanjing is the city where he first saw snow. He got his B.Sc in Computer Science in 1996 and worked in Shanghai for six years as a IT professional. Now he is a Computer Science Master's student in University of Windsor, Canada.