# Together Structures

by

Michael de la Maza

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1997



Author .....
Department of Electrical Engineering and Computer Science
May 23, 1997

Certified by .....
Patrick Henry Winston
Professor
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Departmental Committee on Graduate Students

# Together Structures

by

Michael de la Maza

Submitted to the Department of Electrical Engineering and Computer Science
on May 23, 1997, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

## Abstract

In this thesis, I describe *together structures*, a representation that facilitates the discovery of local pockets of regularity in large, complex databases. An implemented system, based on together structures, unlike traditional systems based on statistics, focuses in on regularities in domains in which most of the data contains little regularity. The together structure representation is expressive, recursive, and memory intensive. In this thesis, I develop the together structure idea and demonstrate the value of the idea using illustrations from problems in information retrieval, financial time series analysis, and chess.

Thesis Supervisor: Patrick Henry Winston
Title: Professor

# Acknowledgments

Thanks to Patrick Winston, my thesis advisor, whose wit and wisdom has not been diminished by fame and fortune.

Thanks to Deniz Yuret who let me dip into his vast sea of talent. He made signal contributions to each of the applications in the thesis. Chapter 2 benefitted from his NPSEG program which segments sentences into noun phrases. Chapter 4 was suggested by his work on a fast move generation algorithm for chess. The work in Chapter 3 has its roots in a conversation that I had with him in 1993. The discussion of the together structure selection function in Chapter 5 is entirely due to him.

Thanks to Randy Davis who provided detail comments on Chapter 3 and to Marvin Minsky whose enthusiasm buoyed the research.

Blake LeBaron reviewed Chapter 3 and made comments that helped persuade my thesis committee that the statistics were done right. Bruce Hansen also reviewed the chapter and found an error

Bistra Dilkina, a high school summer student, implemented the programs and ran the experiments that generated Table 2.2 and Table 2.3. Randy Graebner, a freshman UROP student, implemented the programs and ran the experiments that generated Table 2.4 and Table 2.5.

Thanks to current and former members of the AI Lab Machine learning group including Jonathan Amsterdam, Gary Borchardt, Wes Hildebrandt, Sajit Rao, William Neveitt, and Rick Lathrop.

Charles Isbell kindly let me use Figure 2-27.

And finally thanks to: AA, AB, AC, AD, AE, AF, AG, AH, AI, AJ, AK, AL, AM, AN, AO, AP, AQ, AR, AS, AT, AU, AV, AW, AX, AY, AZ, BA, BB, BC, BD, BE, BF, BG, BH, BI, BJ, BK, BL, BM, BN, BO, BP, BQ, BR, BS, BT, BU, BV, BW, BX, BY, BZ, CA, CB, CC, CD, CE, CF, CG, CH, CI, CJ, CK, CL, CM, CN, CO, CP, CQ, CR, CS, CT, CU, CV, CW, CX, CY, CZ, DA, DB, DC, DD, DE, DF, DG, DH, DI, DJ, DK, DL, DM, DN, DO, DP, DQ, DR, DS, DT, DU, DV, DW, DX, DY, DZ, EA, EB, EC, ED, EE, EF, EG, EH, EI, EJ, EK, EL, EM, EN, EO, EP, EQ, ER,

ES, ET, EU, EV, EW, EX, EY, EZ, FA, FB, FC, FD, FE, FF, FG, FH, FI, FJ, FK, FL, FM, FN, FO, FP, FQ, FR, FS, FT, FU, FV, FW, FX, FY, FZ, GA, GB, GC, GD, GE, GF, GG, GH, GI, GJ, GK, GL, GM, GN, GO, GP, GQ, GR, GS, GT, GU, GV, GW, GX, GY, GZ, HA, HB, HC, HD, HE, HF, HG, HH, HI, HJ, HK, HL, HM, HN, HO, HP, HQ, HR, HS, HT, HU, HV, HW, HX, HY, HZ, IA, IB, IC, ID, IE, IF, IG, IH, II, IJ, IK, IL, IM, IN, IO, IP, IQ, IR, IS, IT, IU, IV, IW, IX, IY, IZ, JA, JB, JC, JD, JE, JF, JG, JH, JI, JJ, JK, JL, JM, JN, JO, JP, JQ, JR, JS, JT, JU, JV, JW, JX, JY, JZ, KA, KB, KC, KD, KE, KF, KG, KH, KI, KJ, KK, KL, KM, KN, KO, KP, KQ, KR, KS, KT, KU, KV, KW, KX, KY, KZ, LA, LB, LC, LD, LE, LF, LG, LH, LI, LJ, LK, LL, LM, LN, LO, LP, LQ, LR, LS, LT, LU, LV, LW, LX, LY, LZ, MA, MB, MC, MD, ME, MF, MG, MH, MI, MJ, MK, ML, MM, MN, MO, MP, MQ, MR, MS, MT, MU, MV, MW, MX, MY, MZ, NA, NB, NC, ND, NE, NF, NG, NH, NI, NJ, NK, NL, NM, NN, NO, NP, NQ, NR, NS, NT, NU, NV, NW, NX, NY, NZ, OA, OB, OC, OD, OE, OF, OG, OH, OI, OJ, OK, OL, OM, ON, OO, OP, OQ, OR, OS, OT, OU, OV, OW, OX, OY, OZ, PA, PB, PC, PD, PE, PF, PG, PH, PI, PJ, PK, PL, PM, PN, PO, PP, PQ, PR, PS, PT, PU, PV, PW, PX, PY, PZ, QA, QB, QC, QD, QE, QF, QG, QH, QI, QJ, QK, QL, QM, QN, QO, QP, QQ, QR, QS, QT, QU, QV, QW, QX, QY, QZ, RA, RB, RC, RD, RE, RF, RG, RH, RI, RJ, RK, RL, RM, RN, RO, RP, RQ, RR, RS, RT, RU, RV, RW, RX, RY, RZ, SA, SB, SC, SD, SE, SF, SG, SH, SI, SJ, SK, SL, SM, SN, SO, SP, SQ, SR, SS, ST, SU, SV, SW, SX, SY, SZ, TA, TB, TC, TD, TE, TF, TG, TH, TI, TJ, TK, TL, TM, TN, TO, TP, TQ, TR, TS, TT, TU, TV, TW, TX, TY, TZ, UA, UB, UC, UD, UE, UF, UG, UH, UI, UJ, UK, UL, UM, UN, UO, UP, UQ, UR, US, UT, UU, UV, UW, UX, UY, UZ, VA, VB, VC, VD, VE, VF, VG, VH, VI, VJ, VK, VL, VM, VN, VO, VP, VQ, VR, VS, VT, VU, VV, VW, VX, VY, VZ, WA, WB, WC, WD, WE, WF, WG, WH, WI, WJ, WK, WL, WM, WN, WO, WP, WQ, WR, WS, WT, WU, WV, WW, WX, WY, WZ, XA, XB, XC, XD, XE, XF, XG, XH, XI, XJ, XK, XL, XM, XN, XO, XP, XQ, XR, XS, XT, XU, XV, XW, XX, XY, XZ, YA, YB, YC, YD, YE, YF, YG, YH, YI, YJ, YK, YL, YM, YN, YO, YP, YQ, YR, YS, YT, YU, YV, YW, YX, YY, YZ, ZA, ZB, ZC, ZD, ZE, ZF, ZG, ZH, ZI, ZJ, ZK, ZL, ZM, ZN, ZO, ZP, ZQ, ZR, ZS,

ZT, ZU, ZV, ZW, ZX, ZY, ZZ. You know who you are.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Overview

In this thesis, I describe *together structures*, a representation that facilitates the discovery of local pockets of regularity in large, complex databases. Using problems from information retrieval, financial time series analysis and chess, I show that an implemented system, based on together structures, focuses in on regularities in domains in which most of the data contains little regularity.

In this chapter, I address the following questions:

- What is database mining and why is it important?
- What are together structures?
- What are some of the properties of together structures?

## 1.1 Background

Uncovering regularities in mountains of data is one of the outstanding problems in the field of artificial intelligence with practical applications in a wide variety of real-world domains.

In this thesis I describe *together structures*, a representation I have devised for capturing co-occurrences. When coupled with a straightforward search procedure, together structures have uncovered performance improving regularities in information

retrieval applications (see Chapter 2), in a stock market application (see Chapter 3), and in a chess game application (see Chapter 4).

## 1.2 Desiderata: What do we want from a database mining system?

Despite the large and growing need for database mining systems, there are few general algorithms that can be applied to this task. database mining algorithms must meet several important criteria before they can be widely applied.

First, database mining algorithms must be fast. Gigabyte sized databases with hundreds of thousands of examples are common in real-world applications so each example can take no more than a few seconds to process. This requirement rules out many concept learning algorithms, such as standard backpropagation neural networks [27] and genetic algorithms [30], because, although they are well suited for small databases, they make many slow, laborious passes through the data and often require several hours to reach equilibrium on databases with just a few thousand instances.

Second, the system must be able to work with many different types of data. Real-world databases are heterogenous, combining both numerical and non-numerical data. Many fast statistical techniques, such as multiple linear regression and principal component analysis, require the data to be expressed in numerical form. Likewise, instance based learning algorithms [3] that use Euclidean distance measures must map non-numerical attributes into a numerical form. If a natural order does not exist for the different values of the non-numerical attribute, then mapping them onto a numerical attribute imposes a false regularity on the data that makes uncovering real regularities more difficult.

Third, database mining systems must be able to uncover a wide variety of regularities. Although every system will have biases, strong *a priori* restrictions on concept spaces are bound to be found limiting when the system is applied to real-world databases. For example, a system that limits itself to uncovering linear relationships will not be able to correctly predict the path of an artillery shot.

Fourth, database mining algorithms must degrade gracefully. Real-world databases are often built by many individuals each of which has a different data entry method. For example, an insurance database may contain reports submitted by many agents, all of whom follow slightly different protocols. A database mining system that requires uniformity across all instances will be unable to adequately handle such databases.

My together structure system achieves some success along each of these four dimensions.

## 1.3 Together structures store co-occurrence information

Together structures store co-occurrence information about domain elements. A *domain element* is either a user-defined abstraction that specifies the level at which inference should take place (called a *primitive domain element* or another together structure. For a vision application, a primitive domain element might be a line segment, a polygon, or a pixel. For a chess application, a domain element might be a piece on a board or a particular configuration of pieces. For a speech recognition application, a domain element might be a morpheme or an utterance.

The together structure system stores co-occurrence information about pairs of domain elements. In a chess application, the system might learn that whenever the white queen and a white rook are on the same row this increases the probability that white will win the game. More formally, each together structure stores five frequencies about a pair of domain elements (call them A and B):

1. The number of times that A and B occur together.
2. The number of times that A occurs and B does not occur.
3. The number of times that A does not occur and B occurs.
4. The number of times that A occurs and it is not known whether B occurs.
5. The number of times that it is not known whether A occurs and B occurs.

**T**($X$,$X$)
**:confirm** $N$
**:disconfirm** $N$ $N$
**:not_confirm** $N$ $N$
where $X$ is a user-defined domain element or a together structure and $N$ is a frequency.

Figure 1-1: Together structure store co-occurence information about domain elements.

| Application | Chapter | Confirm | Disconfirm | Not_confirm |
|---|---|---|---|---|
| Related items | Chapter 2 | Yes | No | No |
| Related terms | Chapter 2 | Yes | No | Yes |
| Stock market | Chapter 3 | Yes | Yes | No |
| Chess | Chapter 4 | Yes | No | No |

Table 1.1: A summary of the frequency information that each application uses.

The first frequency is called *confirm*, the second two frequencies are called *disconfirm*, and the last two frequencies are called *not_confirm*. The syntax of the together structure is shown in Figure 1-1.

These five frequencies permit the direct computation of some conditional probabilities (such as P(A|B)).

Not all applications described in this thesis make use of all of the frequency information. The chess application, described in Chapter 4, only uses the *confirm* numbers, for example. Table 1.1 summarizes what frequency information each application exploits.

## 1.4 Together structures focus in on pockets of regularity

The together structure representation was designed to focus in on pockets of regularity in domains in which there is little regularity. Together structures store frequency information about pairs of domain elements and, thus, create simple, local models of the domain.

Unlike many standard statistical techniques, such as multiple linear regression,

the together structure system does not expend representational capacity on parts of the space that do not contain regularities. This ability is illustrated by the stock market domain, described in Chapter 3. A system that tries to characterize the entire financial time series space will find that its representational capacity has been misused in attempting to capture non-existing regularities. In contrast, an implemented system, based on together structures, focuses in on the parts of the space that do contain regularities and, as a result, the system engineers a profitable trading strategy. Likewise, in the information retrieval domain in which an implemented system based on together structures sifts through megabytes of text to create a thesaurus of related words, the ability to focus in on on a small fraction of the total set of relationships among words is essential. In the chess domain, discussed in Chapter 4, together structures represent partial board positions which have previously been shown to lead to good positions. By focusing in on just these positions, the system does not expend valuable resources attempting to characterize the huge space of all possible chess positions.

## 1.5 Creating a large memory of previous cases minimally means never making the same mistake twice

Every time an example is processed, at least one, and usually many, together structures are updated. The result is a memory of together structures that closely reflects the structure of the data.

In the domain of chess, described in Chapter 4, this means that the system learns to avoid certain aspects of some opening errors (such as advancing pieces too quickly) without having to encounter multiple instances of the same problem. This one-shot learning behavior is shared with Atkeson's memory-based control systems [46] as well as many standard statistical techniques.

## 1.6 Together structures are expressive, recursive, and memory intensive

Together structures have the following properties:

- Order invariant. The order in which data is processed does not change the together structure representation.
- Recursive. Together structures are recursively nested. Together structures can, and often do, contain co-occurrence information about other together structures.
- Memory intensive. The together structure approach to data discovery requires updating a large set of together structures which, even for domains with a few hundred examples, often number in the thousands and require several megabytes of memory to store. To reduce memory requirements, a heuristic procedure focuses on a subset of promising together structures.
- Simple. The together structure representation is very simple. There is only one data type (a domain element) and the inference procedures are limited. This reduces the time required to implement the approach for a particular domain. Despite their simplicity (and, as I will be argue in the thesis, because of it) together structures can support data discovery procedures in a variety of different domains, as illustrated in this thesis.
- Expressive. Within the level of abstraction defined by the domain elements, together structures can express any Boolean circuit. Hence, the type of regularity that can be uncovered in the data is not pre-ordained by the together structure system. Of course, some concepts (in particular, conjunctions of domain elements) are easier to express than others, so choosing a good domain element remains a performance influencing issue.

# Chapter 2

# Application to information retrieval

This chapter describes how together structures were used to improve two information retrieval related applications.

The first section descibes how together structures can be used to add related items to START, a natural language understanding system[35]. These related items, which are automatically synthesized from domain-specific information that is normally provided to START, help guide the user through START's knowledgebase. To see the final output of this program turn to Figure 2-10.

In the second application, described in Section 2.2, the together structure system builds a thesaurus of related terms. These related terms are used to expand queries that are given to an information retrieval system. This expansion enhances the performance of the retrieval system by capturing cross-term similarities. To see how the together structure information retrieval system compares to Inquery, a widely used information retrieval system, turn to Figure 2-24.

## 2.1 Generating related items from parseables is a five step process

This section describes how together structures were used to augment START's ability to guide a user through its knowledgebase. This five step process is summarized in Figure 2-3.

### 2.1.1 START is a natural language understanding system

The START natural language understanding has been under development for a period of eighteen years and has been applied to a diverse set of domains including the Voyager Space Mission, political analysis, and medicine [35]. START's success is measured by its ability to answer questions correctly. All aspects of START, from the parser to the generator, are measured along this dimension. During the first half of 1996, a knowledgebase covering Bosnia was created that includes information about the U.S. mission, geography and climate, and recent events. START/Bosnia can answer hundreds of questions, including:

- What do you know about the rescue of Scott O'Grady?
- How long will the US mission in Bosnia last?
- When did Americans arrive in Bosnia?

One shortcoming of START/Bosnia and START in general is that there is no simple way to navigate through the knowledgebase and no way of discovering which topics fall in START's area of expertise. This shortcoming sometimes leads to question traces like the one shown in Figure 2-1. This question trace is as frustrating for the developers of START as it is for the user: Nothing is worse than knowing that START/Bosnia has the information the user wants, but is unable to access it.

One way of aiding the process of knowledgebase navigation is to guide the user by providing a list of related items that START has in its knowledgebase. Together

structures, when combined with a set of simple data massaging procedures, can automatically generate a list of related items.[1]

### 2.1.2 Criteria of success

For this application of the together structure system there are three criteria of success. First, the developers of START will determine whether the utility of the related items subsystem is great enough to justify their inclusion in the START system. This test is in many ways the highest standard that an artificial intelligence application can meet: Are people who are experts in the area of the application interested in using the AI solution?

Second, the together structure system's related items will be compared to related items returned by the Infoseek web search engine which automatically displays a list of related items whenever it is given a query.

Third, each of the related items will be evaluated by hand to determine whether or not a human would have considered them to be related.

### 2.1.3 Together structures extract related items from parseables

The START/Bosnia knowledgebase can access factual knowledge, images, video clips, databases, World Wide web pages, sound clips, and free text. Some of the factual knowledge is stored in the form of parseables, declarative sentences that can be parsed by START. A few parseables from the START/Bosnia application are shown in Figure 2-2.

Related items are derived from parseables via a five step procedure shown in Figure 2-3

This entire process can be fully automated, although, in practice, because some of the related items are not in fact related, some human pruning takes place in

[1]There is some technology already available for creating related items, but it appears to be quite primitive. I recently asked a popular Web search engine for a map of New York. It suggested the following related topics: "Kuwaiti government," "Restaurants in Maryland," "Earthquakes," "Middle schools in Texas," and "Native American history".

```
==>  HOW MANY US TROUPS ARE THERE IN TUZLA
Parser failed on sentence: (HOW MANY US TROUPS ARE THERE IN TUZLA)
"
April 1, 1996; 4:55:49pm: query received from foo [128.52.38.47]:
"1  "
April 1, 1996; 4:55:50pm: response sent to foo [128.52.38.47]:
"

==>  1

<P>
Sorry, I don't know the answer to your question.<P>
"
April 1, 1996; 4:56:00pm: query received from foo [128.52.38.47]:
"how many troups are there in tuzla  "
April 1, 1996; 4:56:09pm: response sent to foo [128.52.38.47]:
"

==>  HOW MANY TROUPS ARE THERE IN TUZLA

<P>
I don't know the answer. Sorry.<P>

"
April 1, 1996; 4:56:19pm: query received from foo [128.52.38.47]:
"how many soldiers are there in tuzla  "
April 1, 1996; 4:56:20pm: response sent to foo [128.52.38.47]:
"

==>  HOW MANY SOLDIERS ARE THERE IN TUZLA

<P>
Sorry, I don't know the answer to your question.<P>

"
April 1, 1996; 4:56:31pm: query received from foo [128.52.38.47]:
"how many soldiers are there in bosnia  "
April 1, 1996; 4:56:32pm: response sent to foo [128.52.38.47]:
"

==>  HOW MANY SOLDIERS ARE THERE IN BOSNIA

<P>
Sorry, I don't know the answer to your question.<P>

"
April 1, 1996; 4:56:50pm: query received from foo [128.52.38.47]:
"what forces are in bosnia  "
April 1, 1996; 4:56:51pm: response sent to foo [128.52.38.47]:
"
```

Figure 2-1: A question trace that illustrates why related items would make START easier to use and more effective. START cannot answer any of the questions the user asked but can answer the question: "How many troops are there in Bosnia?" Note that START no longer makes this error – "troops" and "soldiers" are now synonyms. Nevertheless, the example serves to illustrate the shortcoming that this work addresses.

step 5. Importantly, no additional data needs to be added to the START/Bosnia knowledgebase. The together structure system builds the related items from data structures that are already present in the application.

### 2.1.4 Creating related items for START/Bosnia

This section walks through the creation of a set of related items for one version of the START/Bosnia parseables database.

Figure 2-4 shows some of the noun phrases produced by the program that segments the parseables data file into noun phrases. This is the output of the first step of the program described in Figure 2-3.

Figure 2-7 shows the 49 noun phrases (out of a total of 165) that appear in more than one sentence. These are the only noun phrases for which related item lists are created.

Figure 2-2 show some of the groupings of the sentences which is the output of the second step of the algorithm shown in Figure 2-3. A total of 165 such groupings, one for each noun phrase, are generated but only 49 groupings, corresponding to the noun phrases in Figure 2-7, generate together structures.

In the third step of the algorithm, the sentences in each grouping are processed by START to form subject-relation-object triples (see Figure 2-5) and then inserted into together structures. Figure 2-8 shows some of the together structures that are created from examining the "Vogosca" grouping of sentences.

Figure 2-9 shows the related items that the together structure subsystem generates when given as input the "war crimes" connected parseable shown in Figure 2-2 and probed with the noun phrase "war crimes." This is the output of the fourth step of the related items algorithm.

The final step is to incorporate the related items into schemas using the :mdlm-related slot by matching noun phrases that appear in the :phrases and :sentences slots. In practice, there is some human intervention at this point. Some irrelevant related items (such as the *somebody* token shown in Figure 2-9) are eliminated. The end result is shown in Figure 2-10.

```
=========
Acquisition
=========
Paul Kaminski is the Under Secretary of Defense for Acquisition and Technology.

=========
It
=========
It is suspected that Zdravko Mucic participated in war crimes.
It is suspected that Hazim Delic participated in war crimes.
It is suspected that Zejnil Delalic-Dedo participated in war crimes.
It is suspected that Goran Lajic participated in war crimes.
It is suspected that Esad Landzo participated in war crimes.
It is suspected that Dusan Tadic participated in war crimes.
It is suspected that Ratko Mladic participated in war crimes.
It is suspected that Radovan Karadzic participated in war crimes.
It is suspected that General Djordje Djukic participated in war crimes.

=========
war crimes
=========
war crimes researcher is the same as a researcher of war crimes.
war crimes committee is the same as a committee for war crimes.
war crimes tribunal is the same as a tribunal for war crimes.
It is suspected that Zdravko Mucic participated in war crimes.
It is suspected that Hazim Delic participated in war crimes.
It is suspected that Zejnil Delalic-Dedo participated in war crimes.
It is suspected that Goran Lajic participated in war crimes.
It is suspected that Esad Landzo participated in war crimes.
Refik Hodzik is a Bosnian war crimes researcher.
It is suspected that Dusan Tadic participated in war crimes.
It is suspected that Ratko Mladic participated in war crimes.
It is suspected that Radovan Karadzic participated in war crimes.
Bekir Gavrankapetanovic is the head of the Bosnian war crimes committee.
It is suspected that General Djordje Djukic participated in war crimes.
```

Figure 2-2: Groups of parseables in the START/Bosnia database that contain, respectively, the noun phrase "Acquisition," "It," and "war crimes". These four noun phrases were selected because they are representative of the groupings that the program makes. Some of the groups, such as the one for "Acquisition," are uninformative because they contain only one sentence. Others, such as "It," are grouped because they have an uninteresting noun phrase in common. Note that a single sentence typically belongs to multiple groupings. For example, "It is suspected that Zdravko Mucic participated in war crimes" belongs to the three groupings: "It," "war crimes," and "Zdravko Mucic."

1. A heuristic procedure (Deniz Yuret's NPSEG program) extracts noun phrases from the parseables. Some of the noun phrases that it extracts are shown in Figure 2-4.

2. For every noun phrase, all of the parseables that contain that noun phrase are grouped together. Some of these groups are shown in Figure 2-2.

3. These groups of parseables are processed by a component of START that converts the parseables into subject-relation-object triples (see Figure 2-5). These triples, which are the domain elements in this application, are stored in together structures.

4. Domain elements that co-occur more than once are considered to be related. The together structures are probed for each noun phrase, and a list of related items (i.e., domain elements that co-occur frequently with that noun phrase) is produced for every noun phrase.

5. The related items are inserted into schemas whenever the noun phrase is found in the :phrases and :sentences slots in the schema.

Figure 2-3: The five step algorithm that processes START/Bosnia's parseables file and produces related items.

### 2.1.5 Results

This section presents the results of the related items system and address whether or not the together structure system meets the criteria of success discussed in Section 2.1.2.

The first criteria of success requires an assessment of the together structure system to be made by START's developer. At the suggestion of START's developer, the use of the related items in START was shifted slightly. Initially, the idea was to match the related items against the user's question and display related items whether or not START was able to answer the question. Instead, the related items are now included in schemas, as shown in Figure 2-11, that are retrieved by START once a question has been parsed. This change has the effect of increasing the probability that the related items list will contain items that are relevant to the question. However, if START cannot answer a question, then no related items are displayed. As a result, the problem illustrated by Figure 2-1 remains. So, although the application does

```
Acquisition
Admiral Leighton Smith
Alija Izetbegovic
American forces
Anthony Lake
Approximately 18400 American troops
Approximately 2 million refugees
Approximately 210000 people
Approximately 383000 people live
Bekir Gavrankapetanovic
Bosnia
Bosnian Serbs
Carl Bildt
Carl Bildt IS
Colonel Aleksa Krsmanovic
Croatia
Defense
Dobrinja
Donald Dugan
Dusan Tadic
Ejup Ganic
Esad Landzo
Franjo Tudjman
GENERAL GEORGE ALFRED JOULWAN
GENERAL WILLIAM W CROUCH
General Djordje Djukic
General Ronald R Fogleman
Goran Lajic
Han Pijesak
.
.
.
vehicle accident
war
war crime
war crimes
war crimes committee
war crimes researcher
war crimes tribunal
war criminal
winters
```

Figure 2-4: Noun phrases from the parseables file. This list shows some of the noun phrases extracted from START/Bosnia's parseables file by Yuret's NPSEG program.

```
==>  REFIK HODZIK IS A BOSNIAN WAR CRIMES RESEARCHER*
*

(|WAR CRIMES RESEARCHER-1| IS BOSNIAN)

(|REFIK HODZIK| IS-A |WAR CRIMES RESEARCHER-1|)



==>  PAUL KAMINSKI IS THE UNDER SECRETARY OF DEFENSE FOR ACQUISITION AND TECHNOLOGY

(|PAUL KAMINSKI| IS-A
 |UNDER SECRETARY OF DEFENSE FOR ACQUISITION AND TECHNOLOGY-1|)
```

Figure 2-5: The START system converts parseables into sets of subject-relation-object triples (for the full structure see Figure 2-6). These triples are then stored in together structures which note how often pairs of triples co-occur. If triples co-occur frequently they are said to be related and appear in each other's related items lists.

```
> (gather '(PAUL KAMINSKI IS THE UNDER SECRETARY OF DEFENSE
            FOR ACQUISITION AND TECHNOLOGY))
*

(|PAUL KAMINSKI| IS-A
 |UNDER SECRETARY OF DEFENSE FOR ACQUISITION AND TECHNOLOGY-1|)
([PAUL KAMINSKI +IS-A UNDER SECRETARY OF DEFENSE FOR ACQUISITION AND TECHNOLOGY-1])
> (describe (first *))
#<Structure LINK 1008096> is a structure of type LINK.
It has 7 slots, with the following values:
 SUBJECT:            |PAUL KAMINSKI|
 RELATION:           IS-A
 OBJECT:             |UNDER SECRETARY OF DEFENSE FOR ACQUISITION AND TECHNOLOGY-1|
 SUBJECT-OF:         ([NIL ?NIL NIL])
 OBJECT-OF:          NIL
 TRUTH-VALUE:        TRUE
 HISTORY-LIST:       (
  #<HISTORY--> { <IS NIL NIL NIL NIL>  126  [NIL ?NIL NIL]  SELF  MAIN })
```

Figure 2-6: START's history structure stores subject, relation, and object information which is extracted for use by the together structure system.

Bosnia
Bosnian Serbs
Carl Bildt
Croatia
Donald Dugan
General Djordje Djukic
General Ronald R Fogleman
Goran Lajic
Han Pijesak
Hazim Delic
Herzegovina
It
Lieutenant General Howell Estes
MAJOR GENERAL WILLIAM LAFAYETTE NASH
NATO
Pfc Floyd E Bright
Radovan Karadzic
Ratko Mladic
Sarajevo
Secretary
State
US
Vogosca
Zdravko Mucic
a Bosnian
a suburb
an American general
chief
commander
deployment
deputy
information
leader
sarajevo
secretary
serbs
state department
state
suburb
the Bosnian Serbs
the President
the commander
the head
the same
the war
the
troops
war crimes researcher
war crimes

Figure 2-7: The 49 noun phrases that appear in more than one sentence. These are the only noun phrases that can have related items. Noun phrases that appear only once in the parseables are judged to occur too infrequently to make co-occurrence assessments.

```
#:|MAYOR-RELATED-TO-VOGOSCA-SUBURB-RELATED-TO-SARAJEVO:9240| is a
symbol.
It has no home package.
Its global value is #S(T PAIRING (MAYOR RELATED-TO VOGOSCA SUBURB
RELATED-TO SARAJEVO)
CONFIRM 2 DISCONFIRM (0 0) NOT_CONFIRM (0 0)).
#:|RAJKO KOPRIVICA-IS-MAYOR-MAYOR-RELATED-TO-VOGOSCA:9241| is a
symbol.
It has no home package.
Its global value is #S(T PAIRING (|RAJKO KOPRIVICA| IS MAYOR MAYOR
RELATED-TO VOGOSCA)
CONFIRM 1 DISCONFIRM (0 0) NOT_CONFIRM (0 0)).
#:|RAJKO KOPRIVICA-IS-MAYOR-VOGOSCA-IS-SUBURB:9248| is a symbol.
It has no home package.
Its global value is #S(T PAIRING (|RAJKO KOPRIVICA| IS MAYOR VOGOSCA
IS SUBURB)
CONFIRM 1 DISCONFIRM (0 0) NOT_CONFIRM (0 0)).
#:|MAYOR-IS-FORMER-VOGOSCA-IS-SUBURB:9255| is a symbol.
It has no home package.
Its global value is #S(T PAIRING (MAYOR IS FORMER VOGOSCA IS SUBURB)
CONFIRM 1 DISCONFIRM (0 0) NOT_CONFIRM (0 0)).
#:|-PURPOSE--VOGOSCA-IS-SUBURB:9261| is a symbol.
It has no home package.
Its global value is #S(T PAIRING ([VOGOSCA +IS-A SUBURB-1] PURPOSE
[*SOMEBODY* +RETURN VOGOSCA] VOGOSCA IS SUBURB)
CONFIRM 1 DISCONFIRM (0 0) NOT_CONFIRM (0 0)).
#:|GOVERNMENT-IS-CENTRAL-VOGOSCA-IS-SUBURB:9266| is a symbol.
It has no home package.
Its global value is #S(T PAIRING (GOVERNMENT IS CENTRAL VOGOSCA IS
SUBURB)
CONFIRM 1 DISCONFIRM (0 0) NOT_CONFIRM (0 0)).
#:|PROVISIONS-RELATED-TO-DAYTON PEACE ACCORD-VOGOSCA-IS-SUBURB:9270|
is a symbol.
It has no home package.
Its global value is #S(T PAIRING (PROVISIONS RELATED-TO |DAYTON PEACE
ACCORD| VOGOSCA IS SUBURB)
CONFIRM 1 DISCONFIRM (0 0) NOT_CONFIRM (0 0)).
#:|SUBURB-IS-SERBIAN-VOGOSCA-IS-SUBURB:9273| is a symbol.
It has no home package.
Its global value is #S(T PAIRING (SUBURB IS SERBIAN VOGOSCA IS SUBURB)
CONFIRM 1 DISCONFIRM (0 0) NOT_CONFIRM (0 0)).
```

Figure 2-8: A sampling of the together structures in which the subject-relation-object triple "Vogosca is suburb" appears.

```
> (find-match-np '|WAR CRIMES|)
(|ZDRAVKO MUCIC| |HAZIM DELIC| |ZEJNIL DELALIC-DEDO| |GORAN LAJIC|
 |ESAD LANDZO| |REFIK HODZIK| |WAR CRIMES RESEARCHER| |DUSAN TADIC|
 |RATKO MLADIC| |RADOVAN KARADZIC| |BEKIR GAVRANKAPETANOVIC|
 *SOMEBODY* |GENERAL DJORDJE DJUKIC| |HAN PIJESAK| HEADQUARTERS
 HEAD |SERB MILITIA| BOSNIAN)
```

Figure 2-9: Noun phrases related to "war crimes".

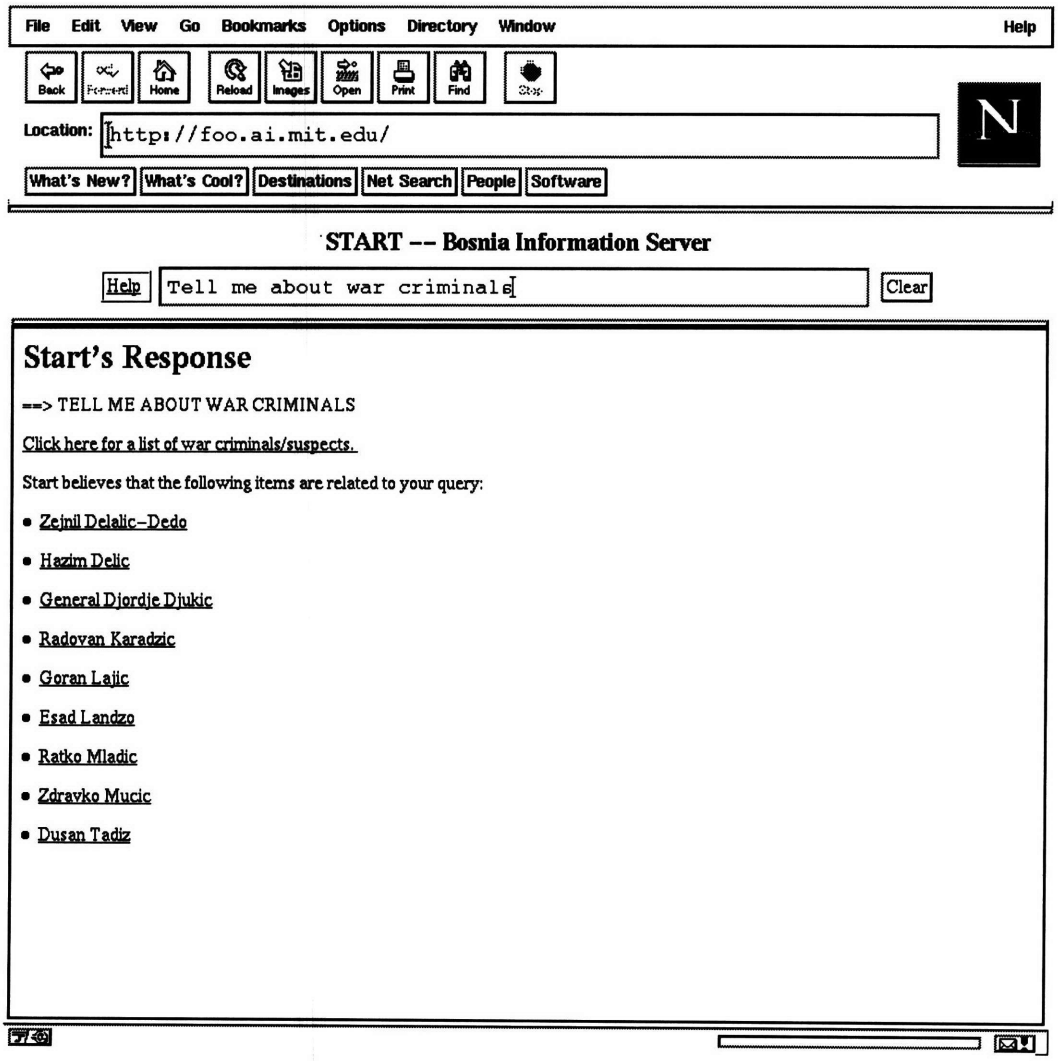


Figure 2-10: A screenshot of START with the related items capability. The START schema that generates this answer is shown in Figure 2-11.

```
(def-schema
  :sentences '("War Crimes Tribunal charged a BOSNIAN-ACTOR with war crimes"
               "War Crimes Tribunal convicted a BOSNIAN-ACTOR"
               "War Crimes Tribunal apprehended a BOSNIAN-ACTOR"
               "War Crimes Tribunal caught a BOSNIAN-ACTOR"
               "Several war criminals remain in Bosnia"
               "Some people have committed war crimes"
               "Some people remain war criminals"
              )
:phrases '(
"several war criminals"
"war crimes"
)
:long-text
'("<P><a
href=\"http://foo.ai.mit.edu/warcriminals.html\">Click
here for a list of war criminals/suspects. </a>
")
:liza '((|WAR CRIMINAL| |WAR CRIME| suspect BOSNIAN-ACTOR)
        (catch convict charge apprehend))
:mdlm-related '("Zejnil Delalic-Dedo" "Hazim Delic"
                "General Djordje Djukic" "Radovan Karadzic" "Goran Lajic"
                "Esad Landzo" "Ratko Mladic" "Zdravko Mucic"
                "Dusan Tadiz")
)
```

Figure 2-11: The *:mdlm-related* slot stores the related items for each schema. When the schema is fired these related items are shown, along with the answer to the question.

meet the first of criteria of success, it does not do so in the way originally envisioned.

The second criteria of success involves comparing the related items list produced by the together structure system to the related items list produced by Infoseek. All 49 of the noun phrases shown in Figure 2-7 were used as queries to Infoseek and the related items were recorded. Figure 2-12 shows some of the related items that were returned by Infoseek. The related items lists stress, for no discernible reason, certain topic areas over other topic areas. For example, in response to the query "Suburb" Infoseek returned three related items, all of which concern Illinois. "Baseball" is also over-represented and Infoseek considers it to be related to both "Leader" (although football, hockey, and basketball are not) and to "US" (although "Government," to choose but one of many examples, is not). So, qualitatively, the related items lists produced by the together structure system are much more sensible than the ones returned by Infoseek. In addition, the Infoseek related items lists failed to return a single relevant related item 58% of the time[2], while the together structure related items list contained at least one item judged to be relevant 100% of the time.

The third of criteria of success requires determining to what degree the noun phrases in the related items lists are relevant. By hand I determined that 65% of the items were relevant and that 83% of the relevant items were in the related items list. Figure 2-13 shows how I judged the related items list for the noun phrase "Vogosca." Because there are no similar systems for automatically generating related items, these figures cannot be easily compared to those of other systems. The most similar algorithms are information retrieval algorithms, such as the ones used in World Wide Web search engines, and these do not perform as well on the task of retrieving relevant articles in response to a query.

## 2.2 Improving a key word retrieval engine

This section describes how together structures were used to improve a key word retrieval engine similar to those that are used by World Wide Web search engines.

[2]I performed the relevance judgments

```
CROATIA
  History of croatia
|MAJOR GENERAL WILLIAM LAFAYETTE NASH|
  <None>
NATO
  Law of Europe
  African governments
SARAJEVO
  Travel guides for croatia
  History of croatia
US
  Baseball
  Baseball in the US
  Java
  Lutheran
VOGOSCA
  Bosnia
SUBURB
  Illinois
  Real estate in Illinois
  Libraries in Illinois
COMMANDER
  Games
  CD-ROM
  Star Trek
DEPUTY
  African governments
  Law
  Department of Defense
LEADER
  U.S. House of Representatives
  Liberal opinions
  Political parties and groups
  Baseball
SERBS
  Travel guides
  Croatia
  History of croatia
```

Figure 2-12: Related items from Infoseek. The query terms submitted to Infoseek are shown in capital letters and the related items returned by Infoseek are shown in an indented list.

- Related items

```
(find-match-np 'vogosca)
(|MUHAMED KOZADRA|
 |RAJKO KOPRIVICA|
 MAYOR
 FORMER
 *SOMEBODY*
 GOVERNMENT
 CENTRAL
 PROVISIONS
 |DAYTON PEACE ACCORD|
 SERBIAN
 FIRST
 SUBURB
 SARAJEVO)
```

- Correct items

```
|MUHAMED KOZADRA|
|RAJKO KOPRIVICA|
MAYOR
GOVERNMENT
|DAYTON PEACE ACCORD|
SERBIAN
SUBURB
SARAJEVO
```

- Incorrect items

```
FORMER
*SOMEBODY*
CENTRAL
PROVISIONS
FIRST
```

Figure 2-13: An example of how the quality of the together structure related items systems is measured. Each element in the first list is considered to be a related item by the system. In this example, out of a total of thirteen related items, eight were judged to be correct and five were judged to be incorrect. Figure 2-14 shows three more examples of related items lists produced by the system.

```
CROATIA::
   |SLOBODAN MILOSEVIC|
   SERBIA
   |FRANJO TUDJMAN|
   PRESIDENT
   SMALLER
|MAJOR GENERAL WILLIAM LAFAYETTE NASH|::
   |PRESIDENT CLINTON|
   |COMMANDER IN CHIEF|
   |ADMIRAL LEIGHTON SMITH|
   NATO
   |LIEUTENANT GENERAL HOWELL ESTES|
   |GENERAL RONALD R FOGLEMAN|
   |GENERAL WILLIAM W CROUCH|
   |GENERAL GEORGE ALFRED JOULWAN|
   AMERICAN
   COMMANDER
   TROOPS
   (IN BOSNIA)
   GENERAL
SARAJEVO::
   LUKAVICA
   |SARAJEVO SUBURBS|
   BOSNIAN
   DOBRINJA
   |MUHAMED KOZADRA|
   VOGOSCA
   |RAJKO KOPRIVICA|
   MAYOR
   FORMER
   |RATKO MLADIC|
   MILITARY
   |RADOVAN KARADZIC|
   LEADER
   POLITICAL
   |MILENKO KARISIK|
   SUBURB
   |DEPUTY INTERIOR MINISTER|
   SERBS
```

Figure 2-14: Related items for three of the forty-nine noun phrases.

In this domain, together structures test the following heuristic:

> *If two words appear close together in the text more often than expected, then these two words can substitute for each other in a query.*

Anecdotally, there are many cases that deny the validity of this heuristic and many cases that support it. For example, the term[3] "new" and the term "york" appear together very often in the phrase "New York," but a system that arbitrarily substituted "new for "york" in a query would probably experience a decline in performance. On the other hand, a document with the term "food" is much more likely to contain references to "restaurants," "eating," and "drinking" than one that does not, and all of these terms might be helpful in expanding the scope of an information request so that it matches documents that might not otherwise be retrieved.

Note that in this application, the together structure system is not locating sets of words that are synonymous with each other. Rather the together structures are storing information about words that are related to each other.

### 2.2.1 Domain description

The test bed for this application was the CACM (Communications of the Association for Computing Machinery) database, a collection of abstracts from the CACM journal, which is a standard test database in the information retrieval field. The CACM database has a total of 3204 documents and 52 natural language queries (three of which are shown in Figure 2-15). Relevance assessments – the list of documents that should be retrieved by each query – have been made by humans.

In addition, the system was tested on several other databases, including a half gigabyte database of *Wall Street Journal* articles, in order to produce collocations[4] for use by START.

[3] A term is a set of characters that lies between two separators. Separators include spaces, commas, apostrophes, and periods.

[4] Collocations are multi-word units that should be treated as single semantic units. "Attorney General" is an example.

- A query composed of key phrases

```
Parallel languages; languages for parallel computation
```

- A query with a story

```
What is the type of a module?  (I don't want the
entire literature on Abstract Data Types here, but
I'm not sure how to phrase this to avoid it. I'm
interested in questions about how one can check that
a module "matches" contexts in which it is used.)
```

- A query with the word "not"

```
Any information on packet radio networks.  Of
particular interest are algorithms for packet
routing, and for dealing with changes in network
topography.  I am not interested in the hardware
used in the network.
```

Figure 2-15: Examples of three queries from the CACM database.

### 2.2.2 Criteria of success

This application of together structures will be judged by the criteria of success in the field of information retrieval. A precision/recall curve will be generated and compared to the precision/recall curve of another information retrieval system on the same data set. *Precision* is the percentage of items that are returned by the system that are relevant. *Recall* is the percentage of relevant items that are returned. For example, if a system returns five documents, two of which are relevant then the precisions is 40% (2/5). If there are a total of four relevant documents then the recall is 50% (2/4).

In addition, because the together structure system produces a list of terms that frequently co-occur together, it can improve a natural language understanding system by noting which multi-term units (e.g., "Attorney General") should be treated as a single semantic unit. Such multi-term units are called collocations.

### 2.2.3 Relevance assessments are sometimes incorrect

Throughout this section and throughout much of the literature on information retrieval, precision/recall graphs, such as the one shown in Figure 2-24, are assumed to be the gold standard for determining success. In this section I give an example of a document in the CACM database that appears to have been mis-classified. This document was correctly retrieved by the together structure system.

The initial query is shown in Figure 2-16. This query is an example of a query composed of key phrases. Note that the first key phrase is "computational complexity."

Figure 2-17 shows the first seventeen documents in the ordered list produced by the together structure system. Article 2110 is judged to be the article that is the most relevant for this query. If only one article were to be returned in response to this query, this is the article that would be returned.

Figure 2-18 lists the eleven articles judged to be relevant by a human. Note that Article 2110 does not appear on this list.

Figure 2-19 shows Article 2110 as it appears in the CACM database. The ".K"

```
computational complexity, intractability, class-complete reductions,
algorithms and efficiency
```

Figure 2-16: A query that illustrates why relevance assessments are sometimes difficult to perform.

field (which stands for "keywords") includes "computational complexity" which is one of the keywords in the original query.

In short, the editor or author of the Article 2110 thought that "computational complexity" was a keyword that should be used to retrieve that article, but whoever performed the relevance assessments that are part of the standard CACM information retrieval database did not. Fortunately, the together structure system correctly, at least in the eyes of the editor/author, retrieved that article.

This example suggests that achieving 100% retrieval accuracy is not possible because there are legitimate disagreements about what articles should be retrieved in response to a query. Hence, information retrieval systems should have the capability to guide the user through sets of related articles. A system for doing just that is discussed in the previous section on related items.

### 2.2.4 Domain element definition

The domain element for this application is a term in an $n$ term text box. Each pair of terms in a text box is said to co-occur. For the experiments described in this chapter, $n$ ranges from 2 to 20.

### 2.2.5 Learning procedure

The learning procedure for this application is straightforward. An $n$ word text box is swept across the database and, at each step, the together structures that correspond to each pair of terms in the text box are updated. A set of *significant* together structures

```
2110.txt 101.750740
3018.txt 91.306519
2997.txt 87.831841
2837.txt 86.578339
3086.txt 82.598618
2986.txt 82.276215
2337.txt 81.493118
2325.txt 79.667740
3165.txt 74.597313
2771.txt 74.498627
2927.txt 74.441055
2702.txt 71.293472
2706.txt 68.911476
3110.txt 67.619080
2703.txt 65.270660
2916.txt 65.028206
2784.txt 64.909454
```

Figure 2-17: The together structures' ranking of the CACM articles in response to the query shown in Figure 2-16. The number to the right of each filename is the score.

```
21 1429
21 1847
21 2189
21 2490
21 2603
21 2701
21 2702
21 2703
21 2932
21 3018
21 3139
```

Figure 2-18: Relevance assessments for the query shown in Figure 2-16. The first column is the query number and the second column indicates the document.

```
An Efficient Context-free Parsing Algorithm
A parsing algorithm which seems to be the most efficient general context-free
algorithm known is described.  It is similar to both Knuth's LR(k) algorithm
and the familiar top-down algorithm.  It has a time bound proportional to
n^3 (where n is the length of the string being parsed) in general; it has a
n^2 bound for unambiguous grammars; and it runs in linear time on a large
class of grammars, which seems to include most practical context-free
programming language grammars.  In an empirical comparison it appears
to be superior to the top-down and bottom-up algorithms studied by Griffiths
and Petrick.
Earley, J.
.K
syntax analysis, parsing, context-free grammar,
compilers, computational complexity
.C
4.12 5.22 5.23
```

Figure 2-19: Article 2110 which is ranked the highest by the together structure system.

which have domain elements that occur significantly more often than expected[5] is selected to participate in the second pass through the data. In the second pass through the database, the significant together structures from the first pass are themselves domain elements and can be paired with primitive domain elements or other together structures. Multiple passes are made through the database until no new significant together structures are discovered.

Once these together structures are created, they are used to expand the queries. An expanded query is created by adding to the query all of the terms that appear in significant together structures with that query. For example, in the CACM database the term "compiler" appears in significant together structures with the terms "macro-processor," "metacompiler," "optimizing," "interpreter," "translator," and "assembler." All of the terms that appear with "compiler" in significant together structures are shown in Figure 2-20. So, if the word "compiler" appears in a query then all of these terms are added to the query with a view towards increasing the probability

[5]For the experiments described in this chapter, "significantly more often" is defined to mean two orders of magnitude more often than expected.

that an exact key word match will retrieve the correct documents. Together structures with more than two elements are critical for retrieving articles such as the one shown in Figure 2-21 which was correctly retrieved in response to the query "SETL, Very High Level Languages" because "high," "level," and "language" appear together in a significant together structure.

### 2.2.6 The scoring procedure weights each term according to its frequency

Once together structures have been created, they are used to expand the query. Terms that appear in significant together structures with terms in the query are added to the query. These terms are called *derived terms.*

Each term in the query has a weight attached to it. Derived terms are given a weight that is determined by the co-occurrence frequencies carried by the together structure. Roughly speaking, the more often the term co-occurs with the term in the original query, the higher its weight. This weight is equal to: $\frac{f_p}{\sqrt{f_1 f_2}}$ where $f_1$ is the frequency of the first term (or together structure) in all of the documents, $f_2$ is the frequency of the second term (or together structure) in all of the documents, and $f_p$ is the frequency of the pair in all of the documents. For the CACM experiments reported in the Section 2.2.7, the window size is 20 so all pairs of terms that appear within 20 terms of each other are said to co-occur.

The weights of the terms in the original query are equal to one plus the weight of the derived term with the maximum weight. The weights of the terms in the original query are always greater than the weights of the derived terms because a match with a term in the original query is thought to be of greater value than a match with a derived term.

All of the terms in the query are then compared against the terms that appear in each document. The score of each document is simply the sum of the weights of all of the terms in the query that appear in the document. This score is then normalized by the length of the document. The higher the score, the better the match with the

```
compiler::

Carter
Zwakenberg
macroprocessor
subsegments
Polish
COUNTESS
Ranelletti
metacompiler
translates
Earley
meta
speeds
optimizing
bootstrapping
Gries
ALCOR
NELIAC
Burroughs
Section
concatenation
typesetting
interpreter
translator
California
Timing
writing
assembler
```

Figure 2-20: All of the terms that appear in significant together structures with the term "compiler".

```
Beyond Programming Languages
As computer technology matures, our growing ability
to create large systems is leading to basic changes
in the nature of programming.  Current programming
language concepts will not be adequate for building
and maintaining systems of the complexity called for
by the tasks we attempt.  Just as high level languages
enabled the programmer to escape from the intricacies
of a machine's order code, higher level programming systems
can provide the means to understand and manipulate complex
systems and components.  In order to develop such systems,
we need to shift our attention away from the detailed
specification of algorithms, towards the description of the
 properties of the packages and objects with which we build.
This paper analyzes some of the shortcomings of programming
languages as they now exist, and lays out some possible
directions for future research.
Winograd, T.
.K
Programming, programming languages, programming systems,
systems development
.C
4.0 4.20 4.22 4.40
```

Figure 2-21: A document that illustrates the importance of creating multi-term together structures. This document was correctly retrieved in response to the query "SETL, Very High Level Languages" because "high," "level," and "language" appear together in a significant together structure.

```
 List all articles dealing with data types in the following languages:
Pascal, CLU, Alphard, Russell, Ada, ALGOL 68, EL1.  List any other languages
that are referenced frequently in papers on the above languages (e.g. catch
any languages with interesting type structures that I might have missed).
```

Figure 2-22: A sample query. The post-processed version of this query is shown in Figure 2-23.

```
List
articles
data
types
languages
Pascal
CLU
Alphard
Russell
Ada
ALGOL
68
EL1
List
languages
referenced
languages
languages
type
structures
```

Figure 2-23: The post-processed version of the query shown in Figure 2-22

query.

### 2.2.7 Results

In Section 2.2.2 I suggested that there are two criteria of success for this application. Determining whether the system meets the first criterion involves computing the system's precision/recall curve and comparing it to that of another system. In the first section below I compare the together structure system's performance on the CACM database with Inquery's performance and find that the precision is indistinguishable for seven of the ten recall points and that the together structure system underperforms the Inquery system on three of the ten recall points. The second criterion is to qualitatively evaluate whether or not the together structure system can locate collocations (multi-term phrases that should be treated as a single syntactic or semantic unit). In the second section below I report experiments on two different types of document collections that suggest that the together structure system can be used to extract collocations. These collocations are now being used to augment START's lexicon.

#### The together structure system outperforms a boolean retrieval system on four of the ten recall points

The results of the together structure information retrieval system are shown in Figure 2-24. This graph compares the together structure system to the Inquery system [68] (which is widely regarded as one of the top information retrieval systems and is the engine behind Infoseek) and to a boolean retrieval system.

A rough heuristic that is commonly found in the information retrieval literature suggests that a 5% difference at a particular recall/precision point is "significant" and a 10% difference is "very significant". Using this heuristic, the Inquery system is significantly better than the together structure system at the 20%, 30%, and 40% recall levels, and the precision is indistinguishable at other recall levels. In particular, at the 10% recall level – the level which most interests us because we are interested in large news article databases were the number of relevant articles is likely to be in

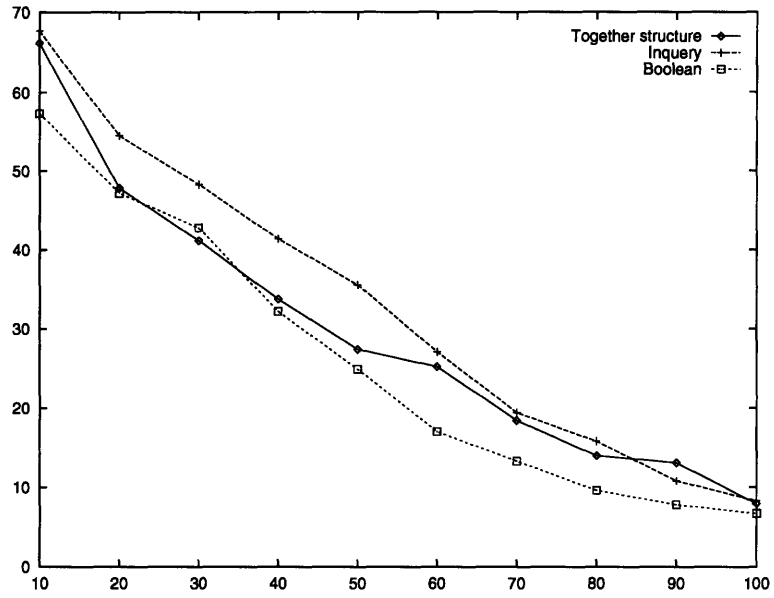


Figure 2-24: A precision/recall graph that compares the performance of the together structure information retrieval algorithm with the performance of the Inquery system [68] system and a boolean system. The dependent variable in this graph is the precision (Y axis) and the independent variable is the recall (X axis). Precision is the percentage of retrieved documents that are relevant and recall is the percentage of relevant documents that are retrieved. An optimal system would achieve 100% precision at each recall level. The together structure system is identical to the boolean system with two exceptions. First, it expands the initial query using terms found in significant together structures and, second, it weights these terms using co-occurrence information stored in the together structures.

the hundreds – the precision of the together structure system and Inquery is virtually identical.

The together structure system is significantly better than the boolean system at the 10%, 60%, 70%, and 90% recall levels, and the performance is indistinguishable at the remaining recall levels.

With two exceptions, the together structure system is algorithmically identical to the boolean system. The boolean system does not expand the terms in the query, and it counts the number of terms in the document that are in the query to form the score. Hence, the boolean system is an internal control on the contribution of the together structures.

Table 2.1 shows important statistics for the CACM database. A total of 9225 together structures are generated. Only 513 terms out of a total of 14008 terms

| | |
|---|---|
| Number of documents | 3204 |
| Average number of terms per document | 64 |
| Number of natural language queries | 52 |
| Number of unique terms | 14008 |
| Number of together structures | 9225 |
| Number of terms in together structures | 513 |

Table 2.1: Statistics for the CACM database.

appear in these together structures. This result is qualitatively consistent with the other two applications described in this thesis: only a small fraction of the domain elements have significant co-occurrences with other domain elements.

### Results on databases with longer documents

In addition to the CACM database, the together structure system has been run on three other databases. Table 2.2 shows two term together structures extracted from Charles Dickens' *A Tale of Two Cities.* This document is markedly different from the CACM database. The writing is not technical and the document is over one thousand times longer than the average document in the CACM database.

Table 2.3 shows two term together structures derived from 2000 news articles taken from the *New York Times*, *USA Today* and *CNN.* This experiment was performed with a window size of 2. The output of this application of together structures is now being used in START to better parse user queries. Note that the system correctly captures multi-word units, such as "Prime Minister" and "New York," that are better treated as a single linguistic unit rather than as adjective-noun phrases.

Table 2.4 and Table 2.5 show, respectively, two and three term together structures drawn from a 500 megabyte database of *Wall Street Journal* articles. The primary purpose of this experiment was to demonstrate that the together structure system scales up to large databases.

| Term1 | Term2 | FPair | FWord1 | FWord2 |
|---|---|---|---|---|
| Madame | Defarge | 122 | 128 | 302 |
| don | t | 99 | 99 | 224 |
| Miss | Pross | 156 | 212 | 162 |
| Saint | Antoine | 50 | 57 | 52 |
| wine | shop | 50 | 119 | 66 |
| La | Force | 18 | 26 | 18 |
| o | clock | 19 | 29 | 27 |
| plane | tree | 15 | 15 | 18 |
| Old | Bailey | 16 | 22 | 20 |
| Charles | Darnay | 54 | 100 | 148 |
| Attorney | General | 14 | 14 | 21 |
| young | lady | 36 | 104 | 56 |
| Jacques | Three | 26 | 73 | 36 |
| Temple | Bar | 13 | 21 | 19 |
| honest | tradesman | 12 | 19 | 16 |
| Doctor | Manette | 80 | 220 | 163 |
| North | Tower | 11 | 17 | 13 |

Table 2.2: Two term together structures from *A Tale of Two Cities* by Dickens. The window size for this experiment was set to 2 so all of these terms are immediately adjacent to each other. The third column, FPair, is the frequency with which the pair of terms appears. The fourth and fifth columns are the frequencies of the single terms.

| Word1 | Word2 | FPair | FWord1 | FWord2 |
|---|---|---|---|---|
| ALL | RIGHTS | 426 | 426 | 426 |
| Air | Force | 263 | 326 | 293 |
| Associated | Press | 962 | 963 | 1000 |
| Bosnian | Serb | 285 | 600 | 387 |
| Boutros | Ghali | 107 | 131 | 107 |
| CNN | Interactive | 424 | 1320 | 424 |
| Cable | News | 430 | 433 | 519 |
| Copyright | Cable | 427 | 1171 | 433 |
| Deve | Gowda | 62 | 62 | 130 |
| EDT | GMT | 383 | 439 | 401 |
| External | sites | 424 | 424 | 528 |
| Golan | Heights | 57 | 81 | 57 |
| Hong | Kong | 175 | 186 | 176 |
| Inc | ALL | 426 | 474 | 426 |
| Khmer | Rouge | 70 | 71 | 76 |
| Los | Angeles | 82 | 92 | 86 |
| McDonnell | Douglas | 55 | 66 | 82 |
| Middle | East | 146 | 164 | 279 |
| NEW | YORK | 54 | 95 | 54 |
| Network | Inc | 427 | 447 | 474 |
| New | York | 918 | 1223 | 920 |
| News | Network | 430 | 519 | 447 |
| North | Korea | 222 | 546 | 412 |
| Northern | Ireland | 182 | 209 | 262 |
| Other | Places | 75 | 193 | 78 |
| Pena | Gomez | 54 | 63 | 65 |
| Prime | Minister | 369 | 373 | 496 |

Table 2.3: Two word together structures extracted from news articles. A total of 2000 news articles from the New York Times, USA Today and CNN were processed with the together structure system. The FPair column shows the frequency of the pair and the FWord1 and FWord2 columns show, respectively, the frequency of the first and second word.

| | | | | |
|---|---|---|---|---|
| wall | street | 133779 | 137308 | 131149 |
| street | journal | 137308 | 124520 | 119910 |
| new | york | 238886 | 94025 | 92787 |
| u | s | 183773 | 991434 | 172087 |
| more | than | 162800 | 135023 | 52540 |
| said | it | 465579 | 452138 | 78493 |
| america | nme | 49538 | 29930 | 29839 |
| will | be | 248557 | 296391 | 61091 |
| north | america | 45191 | 49538 | 31779 |
| united | states | 45060 | 46181 | 30564 |
| year | earlier | 241182 | 53103 | 31592 |
| staff | reporter | 43484 | 24685 | 22887 |
| states | us | 46181 | 49751 | 28843 |
| at | least | 371372 | 23905 | 21761 |
| of | its | 1889328 | 301153 | 49228 |
| real | estate | 31561 | 22223 | 20453 |
| vice | president | 38270 | 102318 | 32423 |
| of | million | 1889328 | 316905 | 46905 |
| chief | executive | 49735 | 54056 | 24072 |
| last | year | 108918 | 241182 | 37240 |
| page | citation | 80101 | 14717 | 14603 |
| stock | exchange | 140780 | 63975 | 23620 |
| amp | co | 138322 | 94905 | 26383 |
| and | chief | 1385887 | 49735 | 21163 |
| he | said | 304330 | 465579 | 43832 |
| he | says | 304330 | 169311 | 30880 |
| this | year | 180940 | 241182 | 34618 |
| journal | wall | 124520 | 133779 | 28279 |
| would | be | 164683 | 296391 | 36801 |
| interest | rates | 57642 | 45332 | 19413 |
| don | t | 36748 | 275294 | 35336 |
| executive | officer | 54056 | 29102 | 16463 |
| tender | offers | 21450 | 23224 | 15258 |
| million | or | 316905 | 187997 | 30572 |
| page | b | 80101 | 43644 | 18686 |

Table 2.4: Two word collocations from the Wall Street Journal. A half gigabyte of Wall Street Journal data was processed to produce this data. The third column is the frequency of the first term, the fourth column is the frequency of the second term, and the fifth column is the frequency with which the first and second term co-occur. Note that some two word collocations are subsets of longer collocations. For example, "wall street" and "street journal" are formed from the longer "wall street journal." This three term collocation, as well as other three term collocations, are also captured by the system, as illustrated by Figure 2.5.

| | | | | | | |
|---|---|---|---|---|---|---|
| wall | street | journal | 133779 | 137308 | 124520 | 119906 |
| street | journal | j | 137308 | 124520 | 116191 | 86775 |
| journal | j | page | 124520 | 116191 | 80101 | 60067 |
| north | america | nme | 45191 | 49538 | 29930 | 29839 |
| united | states | us | 45060 | 46181 | 49751 | 28842 |
| street | journal | wall | 137308 | 124520 | 133779 | 28276 |
| journal | wall | street | 124520 | 133779 | 137308 | 28278 |
| tender | offers | mergers | 21450 | 23224 | 23636 | 15012 |
| offers | mergers | acquisitions | 23224 | 23636 | 26468 | 15011 |
| mergers | acquisitions | tnm | 23636 | 26468 | 25949 | 15011 |
| chief | executive | officer | 49735 | 54056 | 29102 | 16413 |
| j | page | b | 116191 | 80101 | 43644 | 18683 |
| pacific | rim | prm | 21225 | 10192 | 9867 | 9815 |
| consumer | cyclical | cyc | 36616 | 16389 | 9995 | 9970 |
| no | page | citation | 84466 | 80101 | 14717 | 14602 |
| new | york | city | 238886 | 94025 | 26043 | 15028 |
| j | page | c | 116191 | 80101 | 42168 | 13944 |
| j | no | page | 116191 | 84466 | 80101 | 14602 |
| america | nme | united | 49538 | 29930 | 45060 | 10733 |
| york | stock | exchange | 94025 | 140780 | 63975 | 14896 |
| journal | j | no | 124520 | 116191 | 84466 | 14607 |
| wsj | business | brief | 121246 | 86769 | 16477 | 11379 |
| nme | united | states | 29930 | 45060 | 46181 | 10664 |
| cents | a | share | 57825 | 1650990 | 100852 | 27844 |
| j | page | a | 116191 | 80101 | 1650990 | 26766 |
| far | east | fe | 31089 | 19674 | 7302 | 5537 |
| nme | pacific | rim | 29930 | 21225 | 10192 | 5758 |
| staff | reporter | of | 43484 | 24685 | 1889328 | 22647 |
| stock | exchange | composite | 140780 | 63975 | 13693 | 7594 |
| amp | mergers | takeovers | 138322 | 23636 | 7926 | 5601 |
| consumer | non | cyclical | 36616 | 18697 | 16389 | 5125 |
| america | nme | pacific | 49538 | 29930 | 21225 | 5758 |
| exchange | composite | trading | 63975 | 13693 | 68425 | 6659 |
| rim | prm | united | 10192 | 9867 | 45060 | 5690 |
| bond | market | news | 44097 | 152386 | 78285 | 9118 |

Table 2.5: Three word collocations from the Wall Street Journal. A half gigabyte database of Wall Street Journal articles was processed to generate this data. The fourth, fifth, and sixth columns are, respectively, the frequencies of the first, second, and third term. The seventh and final column is the frequency of co-occurrence of all three terms.

```
business::
brief
financing
wsj
small
days
school
international
corp
backed
computer
community
finance
ibm
personal
leaders


currency::
marks
yen
dollar
hard
currency
traders
tokyo
european
german
late
foreign
central
wednesday
meanwhile
tuesday
```

Figure 2-25: Two term together structures from the Wall Street Journal. The first group forms together structures with the term "business" and the second group forms together structures with the term "currency." In this experiment, two words are said to co-occur if they appear within the same twenty word window. The terms are listed in the order in which they are ranked by the utility function. Only the top fifteen terms are shown.

### 2.2.8 Discussion

In this application – which is of enormous practical significance as the result of the growth of the World Wide Web – a simple together structure based system slightly underperforms an *ad hoc* retrieval system. The Inquery system, in addition to having a finely tuned matching function, has access to data that the together structure system does not. For example, a stemmer in the Inquery system collapses inflections into single terms so that, for example, "robot" and "robots" are both considered to be the same term. There is considerable evidence that this stemming improves performance (see, e.g., [29]).

The similar performance of the two systems is a positive comment on the ability of together structures to gather domain relevant information. It lends credence to the idea that storing information about simple co-occurrences can support performance improving inferences. Furthermore, the together structure system produces information – a "thesaurus" of related words – that might prove to be helpful in other information retrieval and language centered applications. The collocations extracted by the together structure system from a half-gigabyte *Wall Street Journal* database are currently being used by the START natural language system.

### 2.2.9 Related work

This section describes four related information retrieval algorithms, Salton's vector space model [59], HNC's MatchPlus [20], the Inquery system [68], and the Latent Semantic Indexing (LSI) approach[16].

Despite important algorithmic differences amongst the four algorithms, their performance is very similar. This suggests that for each query there is a subset of documents that is both relevant and easy to locate and a subset of documents that is both relevant and very difficult to locate. Any reasonable algorithm will be able to retrieve the first set of documents, but no existing algorithm can retrieve the second set of documents.

Table 2.6 summarizes some of the differences amongst these four approaches to

| Algorithm | Dimensionality reduction | Recursive | Representational capacity | Term-based expansion | Inputs |
|---|---|---|---|---|---|
| Inquery/PhraseFinder | N | N | Bounded | N | POS, text |
| LSI | Y | N | Bounded | Y | text |
| Gauch | Y | N | Bounded | Y | text |
| Together structures | N | Y | Arbitrary | N | text |

Table 2.6: Comparison of together structures with other expansion techniques. The first column indicates whether the system reduces the dimensionality of the space. The second column indicates whether the data structures are recursive in nature. The third column indicates whether the representational capacity is bounded (i.e., not Turing equivalent) or unbounded (i.e., Turing equivalent). In the fourth column, POS stands for part of speech.

information retrieval.

### Salton's System

Salton's system is an implementation of the vector space model. The first step of the algorithm is to find all unique terms in the collection of documents. Then the weight of every term in every single document is computed. The term "weight" reflects the term's importance. Larger weight is given to those terms that occur frequently in particular documents, but rarely in other documents. These terms are more important than the others because they are characteristic for these documents and distinguish them from the other. In order to compute the weight, $w_{ik}$, of particular term, $T_k$, in a single document, $D_i$, first the algorithm has to find $tf_{ik}$, the frequency of occurrence of $T_k$ in $D_i$. The next step is to multiply $tf_{ik}$ by $log(N/n_k)$ where $N$ is the size of the document collection, and $n_k$ represents the number of documents in the collection containing $T_k$. This multiplication will reduce the weight of the term if the term occurs in many of the documents because it is no longer characteristic for that particular document.

$$w_{ik} = tf_{ik} \cdot \log\left(N/n_k\right). \tag{2.1}$$

This is not enough because not all documents have an equal chance to be retrieved. The longer documents which have more terms and higher term frequencies

will generate higher document similarities, and therefore will have greater chance to be retrieved. Length normalization is needed; each document vector is made of unit length (in Euclidean distance). Thus the length of the documents does not influence their retrieval potential.

$$w_{ik} = \frac{tf_{ik} \cdot \log(N/n_k)}{\sqrt{\sum_{j=1}^{t}(tf_{ij})^2 \cdot (\log(N/n_j))^2}}. \tag{2.2}$$

For query and document vectors $Q_j = (w_{j1}, w_{j2}, ..., w_{jt})$ and $D_i = (w_{i1}, w_{i2}, ..., w_{it})$ the similarity depends on the proportion and weight of matching terms in the vector. Therefore the similarity function should reflect the similarity between the corresponding term vectors:

$$sim(Q_j, D_i) = \sum_{k=1}^{t} w_{jk} w_{ik}. \tag{2.3}$$

This formula for global similarity suits our purpose very well because if a high term weight, characteristic for the query, is matched with a high term weight in the document as well then this will increase the similarity considerably. The fact that the most characteristic terms have the greatest influence on the level of similarity computed by this formula generates precision in the choice of retrieved information. If the documents match completely then the similarity will be equal to 1 and if they do not match at all (i.e. if no term presented in the query is found in the document) then the similarity will be equal to 0.

A similar procedure occurs on the local level too. The algorithm computes the term weights and the similarity in terms of the sentences in the documents with sufficient global similarity. Then the system shows the user the documents with the highest global and local similarity.

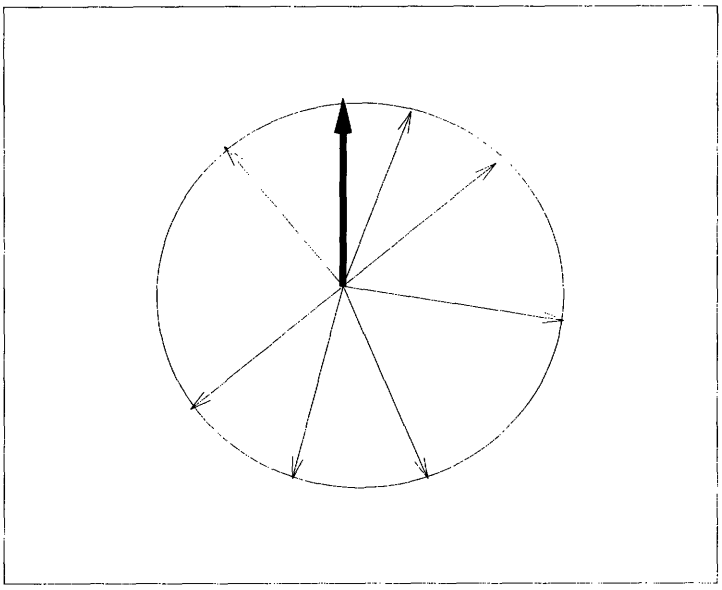

Figure 2-26: Salton's system maps every document and query onto a vector space where each dimension is a term. In this example, the query is the vector with the dotted line and the closest document is the vector marked in bold.

### MatchPlus System

In the MatchPlus system [20] the dimensions of the vector space that represents the words, the documents and the queries are the most frequently used terms[6] in the document collection. These most frequently used terms, which are called features, typically number in the hundreds, although an exact cutoff point is not given.

A neural network algorithm creates the context vector by assigning to every term values for each feature. Term vectors that are close to each other (have similar values for the corresponding features assigned to them) are considered related. The context vector for a document or a query is just the (weighted) sum of the context vectors for those words contained in it. The similarity between the documents and the query is determined by the closeness (in Euclidean distance) of the context vectors by which they are represented.

A critical and possibly fatal assumption made by the system is that words that are used in a similar context have similar meanings. And that all words that have similar meanings are used in a similar context. If this condition does not hold, then

[6]Excluding stop terms such as "and," "the," and "in."

the context vectors will be roughly orthogonal and the "nearness" of the words will not be correctly captured by the system.

At least two commercial systems leverage this technology. The CONVECTIS system automatically indexes and matches free text documents. One novel feature of this system is its ability to use feedback from humans to automatically adjust the context vectors. The Docuverse system converts the context vectors into a graphical representation and gives the user tools to navigate this space.

### Inquery system

The Inquery system [68] uses probabilistic inference networks to combine multiple sources of information and compute the probability that a document matches a query. The Inquery system powers Infoseek, one of the most popular World Wide Web search engines.

The probabilistic inference network approach is an extension of the vector-space model (used by Salton's system, among many others) in which the probability that a document source is relevant is conditioned on knowing a feature (e.g., "What is the probability that a document that contains the term 'computer' should be retrieved when the query contains the term 'robot'?"). After computing such probabilities using simple frequency counts, they are combined using strong independence assumptions. There are two primary advantages of this approach over the vector-space model. First, it provides a straightforward way of combining different information sources. Second, the meaning of the weights has a mathematically rigorous probabilistic interpretation.

### Latent semantic indexing

Dumais and her colleagues [16] describe an approach to information retrieval that attempts to capture similarities amongst terms by representing each term as a feature vector with approximately 100 features. Because all of the terms are mapped onto this space, the feature vectors implicitly capture similarities amongst terms. Similar terms have similar feature vectors.

Each of the 100 features is a linear combination of one of the term-based features.

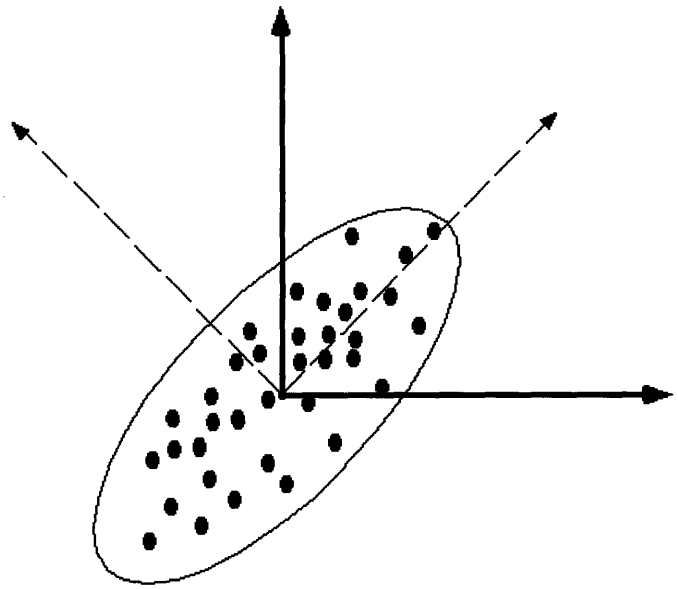

Figure 2-27: An illustration of principal components analysis. The original dimensions of the space are shown in bold and the new principal components dimensions are shown in dotted lines. The dimension along which the data exhibits the most variance is the $y = x$ line and that is the dimension that the principal components analysis will generate first. The Latent Semantic Analysis method chooses approximately the first 100 dimensions generated by PCA.

The linear combination is performed by principal components analysis which selects linear combinations that best capture the variance in the data. Figure 2-27 illustrates this approach on a simple data set.

The LSI approach is very similar to the MatchPlus system. MatchPlus appears to have a more sophisticated and certainly more complicated method of determining the weights of the vectors. Instead of a simple weighting scheme, a neural network is optimized to assign the weights for each context vector.

Tests done with LSI suggest little, if any, performance improvements are obtained over a standard vector-space model in which the full complement of terms is employed. However, because the space has been reduced, the vector multiplication is considerably faster so retrieval times are reduced.

# Chapter 3

# Application to stock market trading

This chapter describes how the together structure system can be applied to uncovering regularities in stock market data.

In the first section of this chapter, I justify why the stock market is a good real-world domain for testing together structures and briefly describe mutual funds, the securities that the together structure strategy trades.

In the second section, I describe how the together structure trading strategy is created and tested. The together structure trading strategy is deemed successful if its performance is statistically better than the performance of a buy-and-hold strategy.

The third section provides evidence that the together structure strategy meets this criteria of success. If you want to know how well the trading strategy performs on out-of-sample data, turn to Figure 3-3. If you want to see how the strategy has performed in real-time trading, turn to Figure 3-4.

In the fourth section I describe experiments that show how the performance of the strategy degrades as together structures are deleted. These experiments show that each together structure has a a small but non-negligible effect on the performance.

In the final section I speculate about why the together structure trading strategy performs better than a buy-and-hold strategy.

## 3.1 The stock market is a good test bed

This section explains why the stock market is a good domain for testing the abilities of the together structure system.

### 3.1.1 Stock market trading is difficult

The following quotations summarize the standard academic view of stock market trading:

> The traditional view of the financial markets is that they are impossible to forecast because they are efficient. All available information is reflected in the current market price.[13]

> The [Random Walk Theory] is based on the reasonable idea that all relevant information is incorporated into current prices without systematic bias, and that individual investors could replicate corporate leverage through their own borrowing and lending. Empirical studies looked at from a distance supported this view. Price movements generated by random numbers look identical to those generated by charts; the performance of analysts and mutual funds is inferior to blind dart throwing; correlations between consecutive changes in individual stocks are close to zero; and the track records of market timers is consistently inferior to a buy-and-hold strategy.[47]

An entire literature provides evidence that no investing strategy can extract above market risk-adjusted returns over an extended period of time. Although this view has come into question recently, a mountain of empirical data indicates that investors, on average, underperform simple buy-and-hold strategies.

As a result, the stock market domain provides a particularly clear and unambiguous test of whether or not the together structure approach can uncover performance improving information.

### 3.1.2 Mutual funds are a good test bed

Out of the many investment securities available in world markets, I chose to run the together structure system on mutual fund data because mutual funds make it possible to ignore some trading details. Mutual funds are investment vehicles managed by professional portfolio managers. These mutual funds typically hold a basket of securities – stocks, bonds, cash – which are selected by the portfolio manager in order to meet certain investment goals.

Many mutual funds provide a daily price, or net asset value, at which investors can purchase or redeem shares in the fund. This pricing gives market participants the ability to trade mutual funds in much the same way that they trade stocks. A host of mutual fund trading strategies (often called timing strategies) have been developed which attempt to outperform the performance of the mutual funds themselves. This chapter describes a together structure based mutual fund timing strategy. The together structure system trades four different funds (Scudder Latin America, Scudder Pacific Opportunities, Scudder Gold, and Scudder Money Market) and, on a daily basis, switches all of its capital into one of the funds.

Creating a strategy for mutual funds as opposed to stocks, futures, or options is preferable for at least two reasons. First, mutual fund companies provide a single price at the end of the day at which all transactions take place. The size of the order, the liquidity of the market, and the speed at which trades are communicated does not effect the execution price. Second, mutual funds have relatively low transaction costs. In particular, the together structure strategy will trade the Scudder mutual funds, a fund family that does not impose any transaction costs when trading in and out of funds. A small fee is charged each day a particular fund family is held.

## 3.2 Creating a together structure trading strategy

In this section, I answer the following questions:

- What is the criteria of success for this application?
- What are the domain elements?
- How does the beam search nest together structures?
- Which together structure is selected to make a prediction?

### 3.2.1 Together structures are tested using standard applied concept learning techniques

The methodology for testing this application of the together structure system is borrowed from the field of applied concept learning (see, e.g., [54]). The available data is split into two sets, the in-sample data set and the out-of-sample data set. The trading strategy is developed on the in-sample data set and tested on the out-of-sample data set. For all of the experiments described in this chapter, the in-sample data begins on September 1, 1988 and ends on December 15, 1992 and the out-of-sample period begins on December 16, 1992 and ends on July 25, 1996.

For this application, the together structure approach will be considered successful if, on the out-of-sample data, its mean daily return is statistically superior to the mean daily return of the buy-and-hold strategy for each of the four funds. The buy-and-hold strategy is the appropriate benchmark because it is equal to the average performance of all investors.[1]

### 3.2.2 Domain elements have three components

One critical step in creating a together structure system for a domain is defining the domain elements. For this application, the domain element specifies a fund, a date relative to the current date, and a percentage change in the fund on that date. The fund is specified by a letter: "P" for the Pacific Opportunities Fund, "L" for the Latin

[1] In fact, the performance of the average investor is slightly worse than buy and hold because of transaction costs.

America Fund, and "G" for the Gold Fund.[2] The date is specified by a non-negative integer which indicates a date prior to the current date (e.g., the number "3" means three days prior to the current date). The percentage change is specified by a positive integer which refers to the bins shown in Table 3.1. For example, the domain element "(P 0 1)" is interpreted as follows:

- The first element in the list, "P", stands for the Pacific Opportunities Fund.
- The second element in the list, "0", refers to today's closing price. A "1" would represent yesterday's closing price and a "5" would represent the closing price five days ago.
- The third element in the list, "1", refers to the first bin shown in Table 3.1.

The fund and date components of the domain element are self-explanatory. The bin number, which is the third component of the domain element, is generated by finding nine values that equally divide the in-sample data. Hence, the days in the in-sample data are equally distributed amongst the ten bins.

A domain element that is true of a particular day is said to *cover* that day. A together structure that is composed of domain elements all of which cover a particular day is said to cover that day. A together structure which covers a particular day and has an accuracy that is no less than that of any other together structure that covers that day is said to be the *best cover* for that day. The accuracy of a together structure is equal to the number of days that the together structure makes a correct prediction divided by the number of days for which it makes a prediction.

### 3.2.3 A beam search procedure nests together structures

As with all together structure applications, a beam search [69] is used to nest together structures. In this domain the number of domain elements is $fdb$ where $f$ is the

[2]The trading strategy can, as indicated in Section 3.1.2, switch into one of of four mutual funds. However, one of these funds, the Scudder Money Market Fund, is a default fund which the trading strategy invests in only when the performance of the other three funds is expected to be poor. Hence, together structures are created for only three funds: the Scudder Pacific Opportunities Fund, the Scudder Latin America Fund, and the Scudder Gold Fund.

| Bin # | Pacific Opportunities | | Latin America | | Gold | |
|---|---|---|---|---|---|---|
| | Left edge | Right edge | Left edge | Right edge | Left edge | Right edge |
| 1 | 0 | 0.993210 | 0 | 0.985437 | 0 | 0.989432 |
| 2 | 0.993210 | 0.996004 | 0.985437 | 0.991678 | 0.989432 | 0.993697 |
| 3 | 0.996004 | 0.997736 | 0.991678 | 0.995290 | 0.993697 | 0.996151 |
| 4 | 0.997736 | 0.999168 | 0.995290 | 0.998033 | 0.996151 | 0.998700 |
| 5 | 0.999168 | 1.000582 | 0.998033 | 1.000627 | 0.998700 | 1.000790 |
| 6 | 1.000582 | 1.001544 | 1.000627 | 1.002912 | 1.000790 | 1.002834 |
| 7 | 1.001544 | 1.003006 | 1.002912 | 1.006463 | 1.002834 | 1.005009 |
| 8 | 1.003006 | 1.004915 | 1.006463 | 1.010296 | 1.005009 | 1.007991 |
| 9 | 1.004915 | 1.008327 | 1.010296 | 1.016265 | 1.007991 | 1.012103 |
| 10 | 1.008327 | Unbounded | 1.016265 | Unbounded | 1.012103 | Unbounded |

Table 3.1: Bins that divide the daily returns of the stock funds. For the Latin America Fund, the first bin contains all of the daily returns less than or equal to 0.985437; the second bin holds all of the daily returns between 0.985437 (non-inclusive) and 0.991678 (inclusive); and the tenth bin contains all of the daily returns that are strictly greater than 1.016265.

number of funds, $d$ is the number of days, and $b$ is the number of bins. For the experiments described in the chapter, $f = 3$, $d = 10$, and $b = 10$ so the total number of domain elements is 300. So there are approximately $300^n$ together structures with $n$ domain elements. Because of the exponential growth of the number of together structures, exhaustively searching all together structures with even three domain elements would be impractical.

The beam search restricts the number of together structures that are generated by eliminating branches in the search tree that are unlikely to lead to good together structures. By greatly reducing the branching factor, the beam search enables the creation of together structures with more domain elements than could be generated by an exhaustive procedure.

This feature does not come without a cost. If a together structure with $l + 1$ domain elements does not have a good precursor with $l$ elements, then the $l + 1$ domain element together structure will never be created because none of the $l$ domain element precursors will be extended. Figure 3-1 gives an example of a case in which the beam search would fail for exactly this reason.

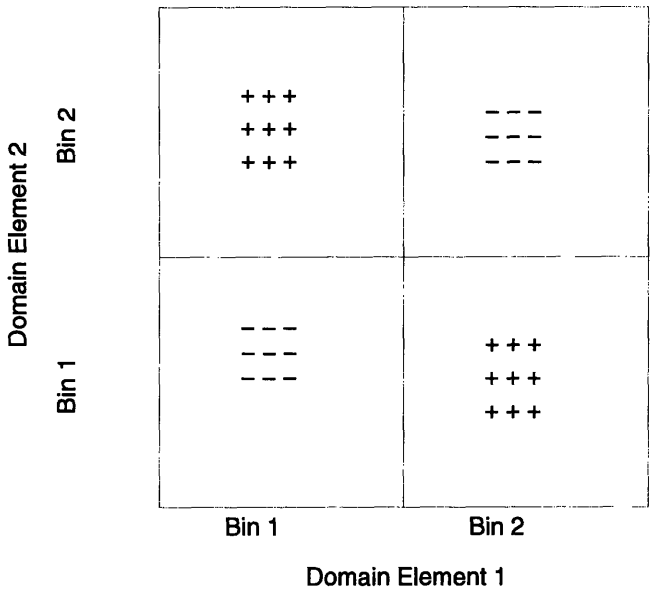


Figure 3-1: A situation that fools the beam search. Dividing the set of examples using either the first or the second domain element is unhelpful. In both cases, the number of positive and negative cases is equal, just as it is in the entire data set. As a result, neither domain element will be extended by the beam search. However, if both domain elements are combined, they divide the space into four segments each of which has only positive or negative instances.

Unlike a traditional beam search, the beam search used for this application does not fix the number of solutions that are extended. Instead, the beam search procedure uses two types of cutoffs to determine which solutions will be killed, kept, or extended. The *quality criterion* determines whether a together structure will be kept. The *extension criterion* determines whether a together structure can be nested in other together structures. No together structure that fails the quality criterion can meet the extension criterion.

For all of the experiments discussed here, the quality criterion compares the number of days which a together structure covers and the number of days which a together structure is expected to cover. If the ratio of these two numbers is above 1.5 then the together structure is kept. The number of days that a together structure is expected to cover is $n * (1/b)^l$ where $n$ is the number of days in the sample, $b$ is the number of bins, and $l$ is the number of domain elements that the together structure contains. The intuition behind this formula can developed by working through an

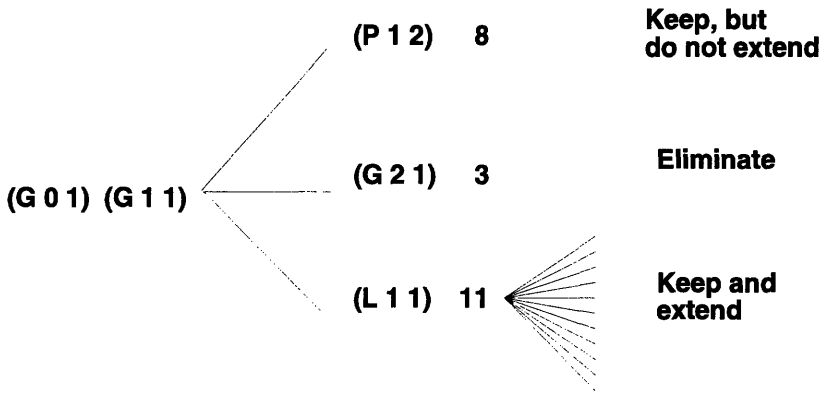


Figure 3-2: Beam search. This figure shows how a two domain element together structure might be extended. The number to the right of the three domain elements in the middle of the figure is the frequency of occurrence of the together structure. The first together structure is kept but not extended because it meets the quality criterion but does not meet the extension criterion. The second together structure is eliminated because it does not meet the quality criterion. The third together structure is both kept and extended.

example. Suppose that a two domain element together structure covers 200 days. If the number of bins is 10, then the expected number of days that an extension of this together structure covers is 20. Hence each additional domain element reduces the number of days that are covered by a factor of $b$. Since the number of days covered by a null together structure (one with no domain elements) is $n$, the number of days in the sample, the expected number of days covered by a together structure is $n * (1/b)^l$.

The extension criterion is a percentage of the total number of days in the sample. If a together structure covers fewer than this percentage of days, then it is not extended. For the experiments described in this chapter, the percentage is set to 1%.

Hence, each together structure can be killed (if it fails the quality criterion), kept (if it passes the quality criterion and fails the extension criterion), or extended (if it passes both the quality and the extension criteria). Figure 3-2 illustrates these three possible outcomes.

Table 3.2 shows how many together structures are generated, kept, and expanded on the in-sample mutual fund data. The vast majority (over 98%) of the together structures that are kept have three domain elements. All of the two domain element together structures are extended while only two of the three domain element together

| Depth | Generated | Kept | Extended |
|---|---|---|---|
| 1 | 9000 | 540 | 540 |
| 2 | 162,000 | 57,923 | 2 |
| 3 | 600 | 140 | 0 |

Table 3.2: Summary of together structure database. The Depth column indicates the depth of the beam search. The number of domain elements in the together structures is one greater than the depth. The Generated column shows how many together structures are generated and tested at that level. The Kept column shows how many of the generated together structures meet the quality criterion and are kept. The Extended column indicates how many of the generated together structures meet the extension criterion. For this data, the quality criterion requires the number of days covered by the together structure to be greater than 50% of what is expected and the extension criterion requires the together structure to cover at least 1% of the total number of days in the sample. The table indicates that at the first level, a total of 9000 together structures are generated, of which 540 pass the quality criterion and the extension criterion. These 540 together structures are expanded into 162,000 together structures, each with three domain elements, of which 57,923 are kept and two of which are expanded. The total number of together structures is 58,603 (540 + 57,923 + 140).

structures are extended.

Table 3.3, Table 3.4, Table 3.5, and Table 3.6 show examples of actual together structures that are generated over the mutual fund data set.[3] Table 3.3 shows the highest frequency two domain element together structures. The first four together structures capture information about extreme movements in the funds. The first together structure, for example, indicates that the worst possible showing in the Gold Fund (a bin 1 performance) is often followed by another worst possible performance. After these first four together structures, many of the rest of the together structures capture co-occurrence information across mutual funds. Some of the domain elements refer to dates as far as seven days back, but none go further than that, suggesting that the ten day window is not much of a restriction.

[3]All of the together structures in this chapter are displayed with parenthesis around each domain element and nowhere else. To recover the original nesting of the together structures, simply right parenthesize the domain elements. For example, the together structure (G 0 1) (G 1 1) (P 1 2) should be read as ((G 0 1) ((G 1 1) (P 1 2))) which means that the together structure ((G 1 1) (P 1 2)) was paired with domain element (G 0 1) by the beam search.

Table 3.4 shows all of the extensions of the first together structure in Table 3.3 which meet the quality criterion and, hence, are kept. All of these together structures store information about how likely it is that the Gold Fund will perform very poorly today (0 days ago). Note that the third domain element is approximately evenly distributed amongst the three mutual funds. Informally, this means that the influence of the three mutual funds on the performance of the Gold Fund is approximately the same once reference has been made to the previous day's movement in the Gold Fund (which is captured by the second domain element in all of the together structures).

Only two three domain element together structures are extended. These are shown in Table 3.5. The first of these two together structures indicates that poor recent performance in the Latin America Fund leads to poor performance in the Pacific Opportunities Fund. Table 3.6 shows the most frequently occurring extensions of the two together structures in Table 3.5.

### 3.2.4 The highest accuracy together structure selects the fund

Once the together structures database has been created, it is run on the out-of-sample data base and the performance is recorded. For each day, the together structure which covers the day and has the highest accuracy in predicting the change in price for the next day is selected.[4] For example, suppose that the following together structures cover a particular day:

1. (L 0 10) (L 1 10) (G 4 7) Accuracy: 56%
2. (P 0 9) (L 1 9) (P 3 2) Accuracy: 57%
3. (G 0 4) (P 3 5) (L 1 2) Accuracy: 63%

The strategy will switch into the Money Market Fund because the highest accuracy together structure predicts that the Gold Fund's performance will be in bin 4 which

[4]Provided that the left edge of the bin is greater than 1. If not, the trading strategy switches into the Scudder Money Market Fund.

```
(G 0  1) (G 1  1) 27
(P 0  1) (P 1  1) 27
(L 0 10) (L 1 10) 25
(L 0  1) (L 1  1) 23
(G 0  4) (P 1  5) 22
(P 0  1) (L 1  1) 22
(L 0  1) (L 4  1) 21
(P 0 10) (P 1  9) 21
(P 0  1) (L 4  1) 21
(L 0 10) (L 3  1) 20
(L 0 10) (P 4  7) 20
(L 0  1) (L 3  1) 20
(P 0  8) (G 3  7) 20
(L 0  7) (G 3  9) 19
(L 0  6) (L 1  5) 19
(P 0  6) (G 2  2) 19
(P 0  4) (G 2  6) 19
(P 0  1) (L 2  1) 19
(G 0 10) (P 1  7) 18
(G 0  9) (L 4  9) 18
(L 0 10) (L 4  1) 18
(L 0  9) (G 4  5) 18
(L 0  7) (G 4  9) 18
(L 0  1) (G 7  8) 18
(L 0  1) (G 4  8) 18
(L 0  1) (G 2  1) 18
(L 0  1) (P 1  1) 18
(P 0 10) (P 2 10) 18
(P 0  8) (L 1  5) 18
(P 0  8) (P 6 10) 18
(P 0  8) (P 4  2) 18
(P 0  6) (G 3  5) 18
(P 0  6) (P 6  5) 18
(P 0  5) (L 5  2) 18
(P 0  2) (G 3  8) 18
(P 0  1) (P 2  1) 18
```

Table 3.3: The highest frequency two domain element together structures. The two lists specify the two domain elements and the right-most number is the frequency of occurrence. For example, the first together structure indicates that a bottom decile move in the Gold Fund was followed by a bottom decile move in the Gold Fund twenty-seven times which is nearly three times the expected frequency.

| | |
|---|---|
| (G 0 1) (G 1 1) (P 1 2) 7 | (G 0 1) (G 1 1) (L 1 1) 8 |
| (G 0 1) (G 1 1) (P 1 5) 5 | (G 0 1) (G 1 1) (L 1 8) 4 |
| (G 0 1) (G 1 1) (P 1 7) 4 | (G 0 1) (G 1 1) (L 2 2) 7 |
| (G 0 1) (G 1 1) (P 2 3) 4 | (G 0 1) (G 1 1) (L 2 3) 4 |
| (G 0 1) (G 1 1) (P 2 5) 5 | (G 0 1) (G 1 1) (L 2 10) 4 |
| (G 0 1) (G 1 1) (P 2 6) 4 | (G 0 1) (G 1 1) (L 3 3) 4 |
| (G 0 1) (G 1 1) (P 2 7) 4 | (G 0 1) (G 1 1) (L 3 4) 6 |
| (G 0 1) (G 1 1) (P 3 3) 4 | (G 0 1) (G 1 1) (L 4 4) 5 |
| (G 0 1) (G 1 1) (P 3 6) 5 | (G 0 1) (G 1 1) (L 4 5) 4 |
| (G 0 1) (G 1 1) (P 3 9) 4 | (G 0 1) (G 1 1) (L 4 9) 4 |
| (G 0 1) (G 1 1) (P 4 6) 4 | (G 0 1) (G 1 1) (L 5 4) 4 |
| (G 0 1) (G 1 1) (P 4 7) 4 | (G 0 1) (G 1 1) (L 5 9) 4 |
| (G 0 1) (G 1 1) (P 4 9) 5 | (G 0 1) (G 1 1) (L 6 5) 6 |
| (G 0 1) (G 1 1) (P 5 6) 4 | (G 0 1) (G 1 1) (L 6 6) 4 |
| (G 0 1) (G 1 1) (P 5 7) 5 | (G 0 1) (G 1 1) (L 6 10) 5 |
| (G 0 1) (G 1 1) (P 5 9) 4 | (G 0 1) (G 1 1) (L 7 5) 5 |
| (G 0 1) (G 1 1) (P 6 7) 5 | (G 0 1) (G 1 1) (L 7 6) 4 |
| (G 0 1) (G 1 1) (P 6 8) 4 | (G 0 1) (G 1 1) (L 8 6) 5 |
| (G 0 1) (G 1 1) (P 6 9) 4 | (G 0 1) (G 1 1) (L 9 2) 5 |
| (G 0 1) (G 1 1) (P 7 1) 4 | (G 0 1) (G 1 1) (L 9 6) 6 |
| (G 0 1) (G 1 1) (P 7 4) 4 | (G 0 1) (G 1 1) (L 9 8) 5 |
| (G 0 1) (G 1 1) (P 7 6) 6 | |
| (G 0 1) (G 1 1) (P 8 1) 4 | |
| (G 0 1) (G 1 1) (P 8 5) 5 | |
| (G 0 1) (G 1 1) (P 8 6) 7 | |
| (G 0 1) (G 1 1) (P 9 1) 5 | |

| | |
|---|---|
| (G 0 1) (G 1 1) (G 2 1) 8 | (G 0 1) (G 1 1) (G 6 8) 4 |
| (G 0 1) (G 1 1) (G 2 2) 6 | (G 0 1) (G 1 1) (G 6 10) 4 |
| (G 0 1) (G 1 1) (G 2 8) 4 | (G 0 1) (G 1 1) (G 7 8) 4 |
| (G 0 1) (G 1 1) (G 3 1) 5 | (G 0 1) (G 1 1) (G 7 9) 5 |
| (G 0 1) (G 1 1) (G 3 2) 4 | (G 0 1) (G 1 1) (G 8 5) 4 |
| (G 0 1) (G 1 1) (G 3 8) 5 | (G 0 1) (G 1 1) (G 8 6) 7 |
| (G 0 1) (G 1 1) (G 4 2) 4 | (G 0 1) (G 1 1) (G 9 2) 4 |
| (G 0 1) (G 1 1) (G 4 8) 6 | (G 0 1) (G 1 1) (G 9 5) 4 |
| (G 0 1) (G 1 1) (G 4 10) 5 | (G 0 1) (G 1 1) (G 9 6) 4 |
| (G 0 1) (G 1 1) (G 5 1) 5 | (G 0 1) (G 1 1) (G 9 7) 4 |
| (G 0 1) (G 1 1) (G 5 8) 6 | (G 0 1) (G 1 1) (G 9 8) 4 |
| (G 0 1) (G 1 1) (G 6 1) 5 | |
| (G 0 1) (G 1 1) (G 6 7) 5 | |

Table 3.4: Extensions of the first together structure shown in Table 3.3. The number to the right of each together structure is the frequency of co-occurrence. The first two domain elements are identical for all of these together structures.

```
(P 0  1) (L 4  1) (L 1  1) 11
(L 0 10) (G 7  1) (L 1 10) 10
```

Table 3.5: The two three domain element together structures that are extended. All other three element together structures fail to meet the criterion for continuing the beam search.

```
(L 0 10) (G 7  1) (L 1 10) (L 7  1) 6
(P 0  1) (L 4  1) (L 1  1) (L 5  1) 5
(P 0  1) (L 4  1) (L 1  1) (G 8  8) 5
(L 0 10) (G 7  1) (L 1 10) (G 6  1) 5
(P 0  1) (L 4  1) (L 1  1) (P 2  1) 4
(P 0  1) (L 4  1) (L 1  1) (L 2  1) 4
(P 0  1) (L 4  1) (L 1  1) (G 6  8) 4
(L 0 10) (G 7  1) (L 1 10) (P 6  2) 4
(L 0 10) (G 7  1) (L 1 10) (L 4  1) 4
(L 0 10) (G 7  1) (L 1 10) (L 3  1) 4
(L 0 10) (G 7  1) (L 1 10) (L 2 10) 4
(P 0  1) (L 4  1) (L 1  1) (P 8 10) 3
(P 0  1) (L 4  1) (L 1  1) (P 8  6) 3
(P 0  1) (L 4  1) (L 1  1) (P 6  4) 3
(P 0  1) (L 4  1) (L 1  1) (P 5  3) 3
(P 0  1) (L 4  1) (L 1  1) (P 4  1) 3
(P 0  1) (L 4  1) (L 1  1) (P 3  3) 3
(P 0  1) (L 4  1) (L 1  1) (P 1  2) 3
(P 0  1) (L 4  1) (L 1  1) (P 1  1) 3
(P 0  1) (L 4  1) (L 1  1) (L 6  6) 3
(P 0  1) (L 4  1) (L 1  1) (L 5  2) 3
(P 0  1) (L 4  1) (L 1  1) (L 3  1) 3
(P 0  1) (L 4  1) (L 1  1) (G 9  8) 3
(P 0  1) (L 4  1) (L 1  1) (G 5  5) 3
(P 0  1) (L 4  1) (L 1  1) (G 3  6) 3
(P 0  1) (L 4  1) (L 1  1) (G 3  5) 3
(P 0  1) (L 4  1) (L 1  1) (G 2  1) 3
```

```
(L 0 10) (G 7  1) (L 1 10) (P 9  4) 3
(L 0 10) (G 7  1) (L 1 10) (P 9  3) 3
(L 0 10) (G 7  1) (L 1 10) (P 8  1) 3
(L 0 10) (G 7  1) (L 1 10) (P 7  1) 3
(L 0 10) (G 7  1) (L 1 10) (P 4  8) 3
(L 0 10) (G 7  1) (L 1 10) (P 3  1) 3
(L 0 10) (G 7  1) (L 1 10) (P 1 10) 3
(L 0 10) (G 7  1) (L 1 10) (L 8  3) 3
(L 0 10) (G 7  1) (L 1 10) (L 6  9) 3
(L 0 10) (G 7  1) (L 1 10) (L 6  1) 3
(L 0 10) (G 7  1) (L 1 10) (L 5  1) 3
(L 0 10) (G 7  1) (L 1 10) (L 2  7) 3
(L 0 10) (G 7  1) (L 1 10) (G 9  8) 3
(L 0 10) (G 7  1) (L 1 10) (G 8  2) 3
(L 0 10) (G 7  1) (L 1 10) (G 8  1) 3
(L 0 10) (G 7  1) (L 1 10) (G 5  1) 3
(L 0 10) (G 7  1) (L 1 10) (G 2  9) 3
(L 0 10) (G 7  1) (L 1 10) (G 2  3) 3
(L 0 10) (G 7  1) (L 1 10) (G 1  9) 3
```

Table 3.6: Extensions of the together structures shown in Table 3.5. The number to the right of each together structure is the frequency of co-occurrence. None of these together structures pass the extension criterion so the iterative beam search procedure is terminated after these together structures are generated.

has a left edge of less than 1. If only the first two together structures were covers, then the strategy would switch into the Pacific Opportunities Fund, as suggested by the second together structure.

Many other selection criteria could have been chosen. For example, choosing the together structure that selects the fund with the highest expected return would be a reasonable alternative. This particular method was chosen because it requires no information beyond what is already contained in the together structures.

The accuracy of a together structure can be computed using the frequency information that is carried by each together structure. The accuracy of a together structure is the number of times that the domain elements co-occur (call this $S$ for success) divided by the number of times that all of the elements except for the domain element that predicts a change in today's price co-occur (call this $T$ for total). In the three examples given above, as with all of the together structures shown in this chapter, the domain element that predicts a change in today's price is always listed first.

The frequency $S$ is equal to the co-occurrence frequency which is carried by every together structure. If $A$ is the domain element that predicts a change in today's price and $B$ is the other domain element (or together structure), $T$ is the frequency of $A$ and $B$ plus the frequency of $\bar{A}$ and $B$. The first frequency is $S$ and the second frequency is equal to one of the two disconfirm frequencies that the together structure carries.

## 3.3 The together structure strategy meets the criteria of success

The first half of this section describes the performance of the together structure trading strategy on the out-of-sample data set. Figure 3-3 is a graph of the cumulative percentage gain on the 3.6 year out-of-sample data set. The second half of this section reports the performance of the trading strategy in real-time trading. Although the real-time data set is far too short to support the claim that the strategy's returns

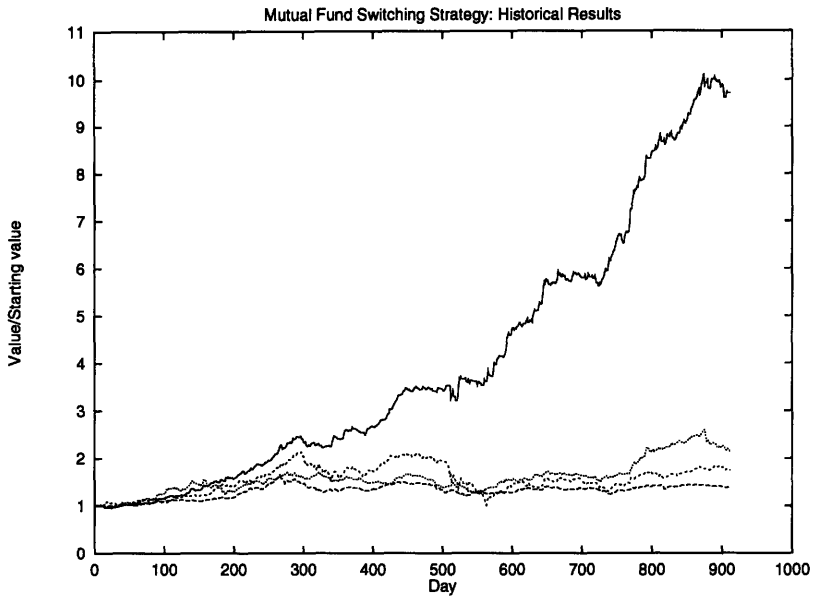


Figure 3-3: The performance of the together structure trading strategy on historical out-of-sample data. This graph shows the return of the together structure trading strategy and the performance of the Scudder Pacific Opportunities Fund, the Scudder Latin America Fund, and the Scudder Gold Fund over the period from December 16, 1992 to July 25, 1996. The graph plots the ratio of the current price to the price on the start date (December 16, 1992) as a function of time. At the right most edge of the graph, the lines correspond to: the together structure trading strategy (top line), the Scudder Gold Fund (second from the top line), the Scudder Latin America Fund (second from the bottom line), and the Scudder Pacific Opportunities Fund (bottom line). The monthly returns of these four time series are given in Table 3.10 and Table 3.11.

will be better than a buy-and-hold strategy, it suggests that the gap between the hypothetical out-of-sample results and actual trading results can be bridged. Figure 3-4 plots the cumulative return of the real-time account.

### 3.3.1 The performance of the together structure strategy is better than buy and hold

This section compares the performance of the together structure strategy with buy-and-hold strategies in each of the three funds. The data suggests that the together structure strategy has a significantly better performance on the out-of-sample data.

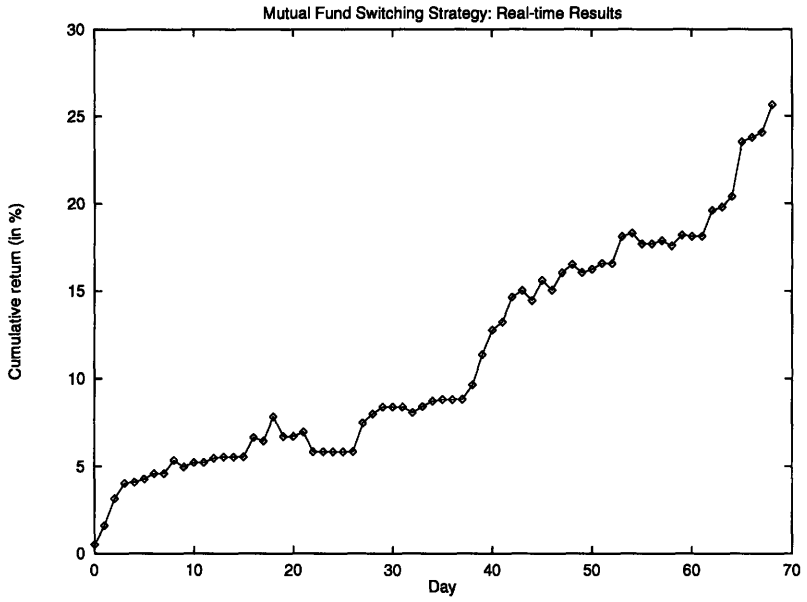


Figure 3-4: The performance of the together structure trading strategy in real time. The graph covers the period from November 7, 1996 to February 14, 1997. In computing the return of the strategy, I assume that the money market fund has a 0% daily return and that there are no transaction costs. Because the return of the money market is higher than the transaction costs (at least for the period shown in the graph), the final cumulative return is slightly higher than the one indicated by the figure. The underlying data used to generate this figure, along with the actual value of the trading account, are shown in Table 3.16 and Table 3.17.

| | Annualized return | Sharpe ratio |
|---|---|---|
| Scudder Pacific Opportunities Fund | 8.62% | 0.36 |
| Scudder Latin America Fund | 16.54% | 0.55 |
| Scudder Gold Fund | 23.94% | 1.06 |
| Together structure strategy | 87.73% | 3.52 |

Table 3.7: Annual return and Sharpe ratio of the Scudder Pacific Opportunities Fund, the Scudder Latin America Fund, the Scudder Gold Fund, and the together structure trading strategy on the out-of-sample period. The Sharpe ratio is the return of the strategy in excess of the T-bill rate divided by the standard deviation of the returns of the strategy.

### Empirical data

Table 3.7 compares the annual returns and Sharpe ratios.[5] The annual return of the together strategy is almost four times greater than the annual return of the best mutual fund. The Sharpe ratio, which is a measure of risk-adjusted return, of the together strategy is more than three times greater than the annual return of the best mutual fund. For comparison purposes, Table 3.8 lists hedge funds with top Sharpe ratios and Table 3.9 lists hedge funds with top annual returns.

Figure 3-3 shows the cumulative profit and loss of the together structure strategy and the three mutual funds over the out-of-sample test period. A cursory inspection of the graph suggest that the together structure strategy performs best when the three mutual funds are rising. This intuition is quantified in Table 3.13 which shows that the cross-correlation of the daily returns of the together strategy and the daily returns of the three mutual funds is high, ranging from a low of 0.50 for the Pacific Opportunities Fund to a high of 0.63 for the Latin America Fund. The cross-correlation with the average return of the three mutual funds is an even higher 0.69.

Table 3.10 and Table 3.11 contain the monthly returns of the three mutual funds and the together structure strategy on the out-of-sample returns. These tables show that the together structure strategy, although it outperforms the three funds by a

[5]The Sharpe ratio is a measure of risk-adjusted return. The Sharpe ratio is equal to $\frac{u-r}{\sigma}$ where $u$ is the mean return of the fund, $\sigma$ is the standard deviation of the returns, and $r$ is the return of a risk-free asset (which is considered to be equal to the return of the US T-bill for this calculation).

| Fund | Sharpe ratio | Annualized return |
|---|---|---|
| III LP | 4.03 | 14.8% |
| Halcyon Institutional | 2.96 | 15.8% |
| Halcyon Arbitrage | 2.88 | 19.3% |
| High Yield Securities Composite | 2.47 | 15.2% |
| Halcyon Distressed Securities | 2.34 | 19.5% |
| GAM Arbitrage Inc. | 2.21 | 13.7% |
| Longfellow Merger Arbitrage | 2.14 | 11.3% |
| Gryphon Hidden Value | 2.05 | 18.4% |
| Halcyon Special Situations | 2.05 | 16.3% |
| Halcyon Private Paper | 1.78 | 17.8% |

Table 3.8: Funds with top Sharpe ratios over a five year period (December 1, 1991 to November 30, 1996) as reported by HedgeMAR [49]. The Sharpe ratio is the mean return of the fund in excess of the T-bill rate of return divided by the standard deviation of the returns. The together structure strategy has a Sharpe ratio of 3.52 over the period 12/16/92-7/26/96. This 3.6 year period is a subset of the five year period covered by the data shown in the table.

| Fund | Annualized return |
|---|---|
| Steven Abernathy Group AG | 48.8% |
| McGinnis Partners Focus Fund | 36.0% |
| Private Investment Fund | 33.3% |
| Puma | 33.3% |
| Tiger | 32.0% |
| Quasar Intl Fund NV | 31.8% |
| TGT Capital Partners | 30.7% |
| Oak Tree Partners | 30.1% |
| Latinvest Fund | 29.5% |
| Roanoke Partners | 29.4% |

Table 3.9: Funds with top annual returns over a five year period (December 1, 1991 to November 30, 1996) as reported by HedgeMAR [49]. The together structure strategy has an annual return of 87.7% over the period 12/16/92-7/26/96. This 3.6 year period is a subset of the five year period covered by the data shown in the table.

significant margin over the long term, sometimes stumbles badly. For example, in the month of June 1993, the return of the together structure strategy is worse than the return of each of the three funds. This means that instead of switching consistently into a high performing fund, the strategy was consistently switching into a low performing fund. Months like these are balanced by periods such as the eight months between April, 1994 and November, 1994 when the together structure strategy outperformed each mutual fund every month.

The together structure strategy is unprofitable during ten of the forty-four months in the out-of-sample period shown in Table 3.10 and Table 3.11. Tellingly, all of the strategy's monthly losses occur during months when the average performance of the three stock funds is negative. This suggests that one way to improve the performance of the together structure trading strategy is to add a negatively correlated fund to the mix.

Even restricting the strategy to the three funds, there is considerable room for improvement. On the out-of-sample data, the strategy chooses the top performing fund only 39.76% of the time and is profitable on 68.4% of the days. There are certain functions, such as moving averages, which are very difficult to compute with the fund/day/bin domain elements. The strategy might benefit from having moving averages, as well as other market indicators, added exogenously as extra time series.

| Date | PO | LA | Gold | Strategy |
|---|---|---|---|---|
| Dec-92 | -0.50% | 2.65% | 0.71% | 0.58% |
| Jan-93 | -0.33% | 0.48% | -1.17% | -3.02% |
| Feb-93 | 3.68% | 2.57% | 6.40% | 4.55% |
| Mar-93 | 2.18% | 7.83% | 9.24% | 4.20% |
| Apr-93 | 4.27% | -4.21% | 9.38% | 8.16% |
| May-93 | 3.34% | 4.31% | 11.18% | 3.47% |
| Jun-93 | -2.20% | 6.90% | 1.76% | 8.47% |
| Jul-93 | 0.60% | 1.77% | 8.57% | 11.28% |
| Aug-93 | 5.07% | 10.75% | -7.21% | 4.29% |
| Sep-93 | 1.21% | 3.43% | -8.79% | 4.88% |
| Oct-93 | 13.76% | 7.29% | 11.45% | 10.71% |
| Nov-93 | 0.61% | 4.89% | -0.82% | 7.41% |
| Dec-93 | 17.48% | 12.63% | 9.69% | 15.43% |
| Jan-94 | -3.10% | 12.41% | 3.83% | 12.36% |
| Feb-94 | -7.14% | -4.31% | -3.47% | -5.31% |
| Mar-94 | -9.91% | -7.59% | 3.07% | -3.01% |
| Apr-94 | 4.01% | -4.13% | -6.09% | 9.69% |
| May-94 | 3.73% | 4.45% | 3.55% | 5.45% |
| Jun-94 | -4.74% | -7.79% | -5.75% | -3.40% |
| Jul-94 | 3.96% | 9.34% | -0.23% | 11.02% |
| Aug-94 | 7.51% | 11.72% | 2.94% | 15.22% |
| Sep-94 | -1.52% | 3.82% | 7.59% | 7.62% |
| Oct-94 | 0.51% | -3.17% | -5.00% | 1.66% |
| Nov-94 | -6.65% | -2.41% | -10.13% | -0.24% |
| Dec-94 | -3.69% | -17.65% | 3.68% | -4.90% |

Table 3.10: Monthly returns for the Scudder Pacific Opportunities Fund, the Scudder Latin America Fund, the Scudder Gold Fund, and the together structure trading strategy on the out-of-sample data. Table 3.13 shows the correlations of these monthly returns. Table 3.7 shows the annual returns of the three stock funds and the together structure trading strategy. The monthly returns of the together structure trading strategy are significantly better ($p < .01$) than the monthly returns of each of the three mutual fund portfolios. Note that the out-of-sample period includes only the latter half of December, 1992 and the first half of July, 1996.

| Date | PO | LA | Gold | Strategy |
|---|---|---|---|---|
| Jan-95 | -7.96% | -10.96% | -11.96% | 10.89% |
| Feb-95 | 4.84% | -12.97% | 1.84% | -3.46% |
| Mar-95 | 0.53% | -0.89% | 11.14% | 16.08% |
| Apr-95 | -0.46% | 11.79% | 4.12% | 11.82% |
| May-95 | 5.74% | 0.74% | 1.72% | 6.35% |
| Jun-95 | 3.74% | 10.23% | 7.52% | 16.32% |
| Jul-95 | -0.42% | -3.95% | -1.35% | 0.12% |
| Aug-95 | -3.13% | 1.56% | -0.53% | 1.46% |
| Sep-95 | 0.62% | -0.74% | -0.67% | -0.10% |
| Oct-95 | -3.40% | -6.95% | -6.04% | -1.41% |
| Nov-95 | -2.12% | 3.64% | 5.67% | 14.05% |
| Dec-95 | 4.19% | 1.30% | 3.06% | 7.39% |
| Jan-96 | 4.93% | 14.22% | 22.12% | 14.64% |
| Feb-96 | 0.60% | -4.56% | 8.45% | 9.33% |
| Mar-96 | -1.14% | 3.42% | 3.08% | 0.34% |
| Apr-96 | 3.33% | 3.52% | 5.02% | 6.03% |
| May-96 | -0.18% | 4.31% | 8.11% | 9.07% |
| Jun-96 | -1.59% | 1.41% | -14.21% | -1.61% |
| Jul-96 | -4.84% | -4.37% | -3.65% | -2.43% |

Table 3.11: Monthly returns for the Scudder Pacific Opportunities Fund, the Scudder Latin America Fund, the Scudder Gold Fund, and the together structure trading strategy on the out-of-sample data. Table 3.13 shows the correlations of these monthly returns. Table 3.7 shows the annual returns of the three stock funds and the together structure trading strategy. The monthly returns of the together structure trading strategy are significantly better ($p < .01$) than the monthly returns of each of the three mutual fund portfolios. Note that the out-of-sample period includes only the latter half of December, 1992 and the first half of July, 1996.

| Fund | % time spent in fund |
|---|---|
| Scudder Pacific Opportunities Fund | 18.95% |
| Scudder Latin America Fund | 35.71% |
| Scudder Gold Fund | 30.67% |
| Scudder Money Market Fund | 14.68% |

Table 3.12: The percentage of time that the together structure trading strategy spends in each of the four funds that it trades amongst. The trading strategy correctly selects the top performing fund 39.76% of the time.

| | PO | LA | Gold | Strategy |
|---|---|---|---|---|
| PO | | 0.51 | 0.46 | 0.50 |
| LA | | | 0.40 | 0.63 |
| Gold | | | | 0.51 |

Table 3.13: Monthly return correlation matrix. The monthly returns of the together structure trading strategy are highly correlated with the monthly returns of each of the three stock funds. The correlation of the average of the monthly returns of the stock funds with the monthly return of the strategy is 0.69.

### Statistical measures of confidence

> The econometric art as it is practiced at the computer terminal involves fitting many, perhaps thousands, of statistical models. One or several that the researcher finds pleasing are selected for reporting purposes. This searching for a model is often well intentioned, but there can be no doubt that such a specification search invalidates the traditional theories of inference.[38]

This section applies a statistical test on the data described immediately above which shows that the performance of the together structure trading strategy is statistically significantly better than buy and hold.

One critical concern in all studies of this sort is to determine whether or not data mining or data snooping has occurred. Because the trading strategy has been fitted to the in-sample data, its performance on the in-sample data is likely to be inflated. Tests show that trading strategies which examine the three most recent daily returns

| Bin # | PO | LA | Gold |
|---|---|---|---|
| 1 | 1 | 3 | 3 |
| 2 | 6 | 9 | 9 |
| 3 | 7 | 10 | 5 |
| 4 | 5 | 8 | 4 |
| 5 | 5 | 10 | 11 |
| 6 | 9 | 7 | 15 |
| 7 | 13 | 6 | 12 |
| 8 | 15 | 10 | 8 |
| 9 | 14 | 14 | 16 |
| 10 | 16 | 14 | 8 |

Table 3.14: Thirty together structures of the form (P 1 10) (*fund* 0 *bin*). The *fund* slot of the second domain element corresponds to the fund and is listed in the first row and the *bin* slot which corresponds to the bin number is listed in the first column. Hence the upper left hand together structure is (P 1 10) (P 0 1) and it stores how many times a top decile move in the Pacific Opportunities Fund is followed on the next day by a bottom decile move in the Pacific Opportunities Fund. This table shows that a top decile move in the Pacific Opportunities Fund is followed by strong moves in the Pacific Opportunities Fund, the Latin American Fund, and the Gold Fund more often than would be expected. For the Pacific Opportunities Fund, the number of occurrences in the last three bins is 45 (15+14+16) which is 64.8% more than would be expected if the occurrences were evenly distributed throughout the ten bins. For the Latin America Fund, the number of occurrences in the last three bins is 38 (10+14+14) which is 39.2% more than would be expected if the occurrences were evenly distributed throughout the ten bins. For the Gold Fund, the number of occurrences in the last three bins is 32 (8+16+8) which is 17.2% more than expected.

| Bin # | PO | LA | Gold |
|---|---|---|---|
| 1 | 0 | 1 | 2 |
| 2 | 1 | 2 | 0 |
| 3 | 1 | 2 | 2 |
| 4 | 1 | 1 | 0 |
| 5 | 3 | 2 | 1 |
| 6 | 0 | 1 | 2 |
| 7 | 0 | 1 | 2 |
| 8 | 4 | 2 | 3 |
| 9 | 3 | 2 | 2 |
| 10 | 3 | 2 | 2 |

Table 3.15: Thirty together structures of the form (P 1 10) (P 0 10) (*fund* 2 *bin*). The *fund* slot of the second domain element corresponds to the fund and is listed in the first row and the *bin* slot which corresponds to the bin number is listed in the first column. Hence the lower left hand together structure is (P 1 10) (P 0 1) (P 2 10) and it stores how many times a bottom decile move in the Pacific Opportunities Fund is preceded by two top decile moves in a row in the Pacific Opportunities Fund. These together structures are extensions of the together structure in the lower left hand corner of Table 3.14. This table shows that two top decile moves in a row in the Pacific Opportunities Fund are followed more often than expected by strong moves in the three stock funds. In the Pacific Opportunities fund the frequency in the top three deciles is 108% greater than expected, in the Latin America Fund the frequency is 25% greater than expected, and in the Gold Fund the frequency is 46% greater than expected.

of a time series can show extraordinary profits on in-sample data [70].

The best way to address this issue is not to apply finely tuned data mining corrections, but rather to segregate a clean out-of-sample data set on which the strategy is run only once, after it has been developed on the in-sample data. The approach of maintaining a clean out-of-sample data set, as I have done, is the "only fully satisfactory solution" [72] to addressing questions of statistical significance and data mining. The statistical test described here is performed on the out-of-sample data set.

This statistical test measures whether or not two sample distributions are drawn from the same distribution. This test shows that the difference in mean daily returns between the together structure strategy and a buy-and-hold strategy in each of the funds is significant ($p < .01$). The p value is obtained from a two sample t-test that assumes unequal variances. The t value is given by:

$$\frac{x_1 - x_2}{\sqrt{\frac{\sigma_1^2+\sigma_2^2-2*Cov(x_1,x_2)}{n}}} \tag{3.1}$$

where $x_1$ and $x_2$ are the mean daily returns of the two strategies, $\sigma_1$ and $\sigma_2$ are the standard deviations of the daily returns of the two strategies, $n$ is the number of samples, and $Cov(x_1, x_2)$ is the covariance of $x_1$ and $x_2$.

### 3.3.2 The together structure strategy is a trend-following strategy

Broadly speaking, the together structure strategy appears to locate trends in the three mutual funds and switch to the fund that has the highest probability of following the trend. This is in contrast to strategies which attempt to buy bottoms and sell tops. The together structure strategy instead locates a fund that has already started to move upwards, assesses the probability that it will continue to move upwards on the next day, and then selects the fund that has the highest probability of showing a profit on the next day.

This is illustrated by Table 3.14 and Table 3.15. Table 3.14 shows that strong upward moves in the Pacific Opportunities Fund are preceded by strong moves in the

fund more often than expected. Curiously, they are also preceded by strong moves in the Latin America Fund and in the Gold Fund more often than expected. Table 3.15 shows all of the possible extensions of one of the together structures in Table 3.14 and indicates that strong moves in the Pacific Opportunities Fund are preceded by two consecutive strong moves in the fund more often than expected. As was the case with the together structures in Table 3.14, this phenomena is also true of the Latin America Fund and the Gold Fund.

### 3.3.3 The together structure strategy performs well in real-time trading

Starting on November 7, 1996 and continuing to this date a small six digit account has been traded in real time using the together structure trading strategy. Table 3.16 and Table 3.17 give detailed data about the performance of this trading account, including daily returns. Figure 3-4 charts the daily cumulative return of the strategy.

The performance of the fund during the real-time test period has been above what would be expected from the out-of-sample test. However, this should be discounted because the average return of the three mutual funds has been above its long run average during this period. Over the real-time period (11/7/96-2/14/97), the Pacific Opportunities Fund rose from 16.21 to 17.57, a gain of 8.38%; the Latin America Fund rose from 20.87 to 25.27, a gain of 21.08%; and the Gold Fund fell from 13.61 to 13.01, a loss of 4.41%.[6] The return of the Pacific Opportunities Fund over this three and a half month period is just short of its *annual* return over the out-of-sample period (as shown in Table 3.7) and the return of the Latin America Fund is in fact higher than its annual return. Note that despite the above average performance of those two funds, the together structure strategy still outperformed (although not by a statistically significant amount) a buy-and-hold strategy in each of the three funds.

More importantly, this real-time test period served to verify that the trading strategy can actually be executed. Four critical concerns were addressed. First, the

[6] All of these figures are adjusted for yearly distributions which took place on December 27, 1996.

transaction costs, although not zero as had been assumed in out-of-sample testing, were minimal. The transaction costs amounted to less than 0.4% of the principal in the account. This entire cost was charged to the account in December when a foreign tax was applied to a trade in the Latin America Fund. Second, the trading has not been restricted in any way, despite the fact that the frequency of trading is very high. Third, the real-time trading requires computing the closing price of the three funds slightly prior to the close so that a trade can be entered on that day. Fortunately, mutual funds disclose their holdings on a quarterly basis and this information, along with the price of each holding a few minutes prior to the closing, can be used to estimate the closing price. Fourth, although there have been human errors during the real-time period which have resulted in incorrect trades they have not outweighed the edge provided by the together structure strategy.

## 3.4 The performance degrades gracefully

This section describes experiments that test how the performance of the together structure strategy changes when the set of together structures is modified. The purpose of these tests is to understand, to some degree, why the together structure system achieves the performance described in the previous section.

The nesting of the together structures automatically adds a degree of robustness to the final set of the together structures. If a single together structure is deleted, then the performance is unlikely to deteriorate because either a parent of the together structure or a child of the together structure will seamlessly replace the deleted together structure.

Of course, this seamless replacement can only happen up to a point. Figure 3-5 quantifies how quickly the performance degradation takes place. Impressively, all of the 253 together structures (see Table 3.20 for a list of the most frequently occurring ones) that are the best covers on the out-of-sample data set can be eliminated and the annual return declines by a statistically insignificant amount.

Figure 3-6 explains, in part, why the together structure system is so robust. On

| Date | Daily return | Cumulative return | Buy price | Sell price | Fund | Estimated value | Additions | Actual value |
|---|---|---|---|---|---|---|---|---|
| 11/07/96 | 0.514% | 0.514% | 13.62 | 13.69 | Gold | $101,111.00 | $100,594.00 | N/A |
| 11/08/96 | 1.096% | 1.615% | 13.69 | 13.84 | Gold | $102,218.87 | $0.00 | N/A |
| 11/11/96 | 1.517% | 3.157% | 13.84 | 14.05 | Gold | $103,769.88 | $0.00 | N/A |
| 11/12/96 | 0.854% | 4.038% | 14.05 | 14.17 | Gold | $104,656.17 | $0.00 | $104,656.00 |
| 11/13/96 | 0.048% | 4.088% | 20.96 | 20.97 | LA | $104,706.10 | $0.00 | $104,706.00 |
| 11/14/96 | 0.181% | 4.276% | 16.55 | 16.58 | PO | $104,895.90 | $0.00 | N/A |
| 11/15/96 | 0.302% | 4.591% | 16.58 | 16.63 | PO | $105,212.23 | $0.00 | $105,212.00 |
| 11/18/96 | 0.000% | 4.591% | 1.00 | 1.00 | MM | $105,212.23 | $0.00 | $105,225.00 |
| 11/19/96 | 0.721% | 5.345% | 16.65 | 16.77 | PO | $105,970.52 | $0.00 | N/A |
| 11/20/96 | -0.358% | 4.968% | 16.77 | 16.71 | PO | $105,591.37 | $0.00 | $105,604.00 |
| 11/21/96 | 0.240% | 5.220% | 20.84 | 20.89 | LA | $105,844.71 | $0.00 | N/A |
| 11/22/96 | 0.000% | 5.220% | 20.89 | 20.89 | LA | $105,844.71 | $0.00 | $105,858.00 |
| 11/25/96 | 0.237% | 5.469% | 16.86 | 16.90 | PO | $106,095.83 | $0.00 | N/A |
| 11/26/96 | 0.059% | 5.532% | 16.90 | 16.91 | PO | $106,158.60 | $0.00 | N/A |
| 11/27/96 | 0.000% | 5.532% | 1.00 | 1.00 | MM | $106,158.60 | $0.00 | $106,171.00 |
| 11/29/96 | 0.000% | 5.532% | 1.00 | 1.00 | MM | $106,158.60 | $0.00 | N/A |
| 12/02/96 | 1.051% | 6.641% | 20.93 | 21.15 | LA | $107,274.46 | $0.00 | N/A |
| 12/03/96 | -0.189% | 6.439% | 21.15 | 21.11 | LA | $107,071.58 | $0.00 | $107,137.00 |
| 12/04/96 | 1.293% | 7.815% | 13.15 | 13.32 | Gold | $108,455.77 | $0.00 | $108,522.00 |
| 12/05/96 | -1.033% | 6.702% | 21.30 | 21.08 | LA | $107,335.57 | $0.00 | $107,402.00 |
| 12/06/96 | 0.000% | 6.702% | 1.00 | 1.00 | MM | $107,335.57 | $0.00 | N/A |
| 12/09/96 | 0.237% | 6.955% | 21.08 | 21.13 | LA | $107,590.16 | $0.00 | $107,656.00 |
| 12/10/96 | -1.047% | 5.835% | 13.37 | 13.23 | Gold | $106,463.57 | $0.00 | $106,542.00 |
| 12/11/96 | 0.000% | 5.835% | 1.00 | 1.00 | MM | $106,463.57 | $0.00 | N/A |
| 12/12/96 | 0.000% | 5.835% | 1.00 | 1.00 | MM | $106,463.57 | $0.00 | N/A |
| 12/13/96 | 0.000% | 5.835% | 1.00 | 1.00 | MM | $106,463.57 | $0.00 | N/A |
| 12/16/96 | 0.000% | 5.835% | 1.00 | 1.00 | MM | $106,463.57 | $0.00 | N/A |
| 12/17/96 | 1.537% | 7.462% | 20.82 | 21.14 | LA | $108,099.89 | $0.00 | N/A |

Table 3.16: The trades over the real-time test period: November 7, 1996 to December 17, 1996. The first column is the date. The second column shows the daily return. The third column shows the inception to date return or cumulative return. The fourth and fifth columns are, respectively, the buy and sell prices. The sixth column indicates which mutual fund was traded (LA = Scudder Latin America Fund; PO = Scudder Pacific Opportunities Fund; Gold = Scudder Gold Fund; MM = Scudder Money Market Fund). The seventh column shows the estimated value of the account. The eighth column shows additions to the principal in the account. The ninth column shows the actual value of the account, as indicated by the brokerage firm. The actual capital in the account differs from the estimated capital for two reasons. First, the estimated amount assumes that the return of the money market fund is 0% when, in fact, it is slightly positive. Second, the estimated amount assumes that there are no costs associated with trading while the actual amount reflects these trading costs.

| Date | Daily return | Cumulative return | Buy price | Sell price | Fund | Estimated value | Additions | Actual value |
|---|---|---|---|---|---|---|---|---|
| 12/18/96 | 0.473% | 7.970% | 21.14 | 21.24 | LA | $108,611.25 | $0.00 | $108,770.00 |
| 12/19/96 | 0.377% | 8.377% | 21.24 | 21.32 | LA | $109,020.33 | $0.00 | N/A |
| 12/20/96 | 0.000% | 8.377% | 1.00 | 1.00 | MM | $109,020.33 | $0.00 | N/A |
| 12/23/96 | 0.000% | 8.377% | 21.36 | 21.36 | LA | $109,020.33 | $0.00 | N/A |
| 12/24/96 | -0.301% | 8.051% | 16.63 | 16.58 | PO | $108,692.55 | $0.00 | N/A |
| 12/26/96 | 0.325% | 8.402% | 21.51 | 21.58 | LA | $109,046.26 | $0.00 | N/A |
| 12/27/96 | 0.281% | 8.707% | 21.32 | 21.38 | LA | $109,353.15 | $0.00 | N/A |
| 12/30/96 | 0.094% | 8.809% | 21.38 | 21.40 | LA | $109,455.44 | $0.00 | $109,626.00 |
| 12/31/96 | 0.000% | 8.809% | 1.00 | 1.00 | MM | $109,455.44 | $0.00 | N/A |
| 01/02/97 | 0.000% | 8.809% | 1.00 | 1.00 | MM | $109,455.44 | $0.00 | $109,680.00 |
| 01/03/97 | 0.772% | 9.649% | 16.85 | 16.98 | PO | $110,299.91 | $0.00 | $110,626.00 |
| 01/06/97 | 1.568% | 11.368% | 21.68 | 22.02 | LA | $112,029.70 | $0.00 | N/A |
| 01/07/97 | 1.272% | 12.784% | 22.02 | 22.30 | LA | $113,454.24 | $0.00 | $113,649.00 |
| 01/08/97 | 0.404% | 13.239% | 22.30 | 22.39 | LA | $113,912.13 | $0.00 | $114,146.00 |
| 01/09/97 | 1.249% | 14.654% | 12.01 | 12.16 | Gold | $115,334.84 | $0.00 | N/A |
| 01/10/97 | 0.329% | 15.031% | 12.16 | 12.20 | Gold | $115,714.23 | $0.00 | $115,952.00 |
| 01/13/97 | -0.492% | 14.465% | 12.20 | 12.14 | Gold | $115,145.15 | $0.00 | N/A |
| 01/14/97 | 1.000% | 15.610% | 23.00 | 23.23 | LA | $116,296.60 | $0.00 | $116,535.00 |
| 01/15/97 | -0.474% | 15.062% | 23.23 | 23.12 | LA | $115,745.90 | $0.00 | N/A |
| 01/16/97 | 0.865% | 16.058% | 23.12 | 23.32 | LA | $116,747.17 | $0.00 | $116,987.00 |
| 01/17/97 | 0.386% | 16.506% | 23.32 | 23.41 | LA | $117,197.73 | $0.00 | $117,438.00 |
| 01/20/97 | -0.396% | 16.044% | 17.66 | 17.59 | PO | $116,733.19 | $0.00 | N/A |
| 01/21/97 | 0.171% | 16.242% | 17.59 | 17.62 | PO | $116,932.28 | $0.00 | N/A |
| 01/22/97 | 0.284% | 16.572% | 17.62 | 17.67 | PO | $117,264.10 | $0.00 | $117,504.00 |
| 01/23/97 | 0.000% | 16.572% | 1.00 | 1.00 | MM | $117,264.10 | $0.00 | N/A |
| 01/24/97 | 1.319% | 18.109% | 12.13 | 12.29 | Gold | $118,810.86 | $0.00 | $119,055.00 |
| 01/27/97 | 0.163% | 18.301% | 12.29 | 12.31 | Gold | $119,004.21 | $0.00 | N/A |

Table 3.17: The trades over the real-time test period: December 18, 1996 to January 27, 1997. The first column is the date. The second column shows the daily return. The third column shows the inception to date return or cumulative return. The fourth and fifth columns are, respectively, the buy and sell prices. The sixth column indicates which mutual fund was traded (LA = Scudder Latin America Fund; PO = Scudder Pacific Opportunities Fund; Gold = Scudder Gold Fund; MM = Scudder Money Market Fund). The seventh column shows the estimated value of the account. The eighth column shows additions to the principal in the account. The ninth column shows the actual value of the account, as indicated by the brokerage firm. The actual capital in the account differs from the estimated capital for two reasons. First, the estimated amount assumes that the return of the money market fund is 0% when, in fact, it is slightly positive. Second, the estimated amount assumes that there are no costs associated with trading while the actual amount reflects these trading costs.

| Date | Daily return | Cumulative return | Buy price | Sell price | Fund | Estimated value | Additions | Actual value |
|---|---|---|---|---|---|---|---|---|
| 01/28/97 | -0.522% | 17.684% | 22.98 | 22.86 | LA | $118,382.78 | $0.00 | N/A |
| 01/29/97 | 0.000% | 17.684% | 1.00 | 1.00 | MM | $118,382.78 | $0.00 | N/A |
| 01/30/97 | 0.160% | 17.872% | 12.49 | 12.51 | PO | $118,572.34 | $0.00 | N/A |
| 01/31/97 | -0.240% | 17.590% | 12.51 | 12.48 | PO | $118,287.99 | $0.00 | N/A |
| 02/03/97 | 0.508% | 18.187% | 23.62 | 23.74 | LA | $118,888.95 | $0.00 | N/A |
| 02/04/97 | -0.057% | 18.119% | 17.48 | 17.47 | PO | $118,820.93 | $0.00 | N/A |
| 02/05/97 | 0.000% | 18.119% | 1.00 | 1.00 | MM | $118,820.93 | $0.00 | N/A |
| 02/06/97 | 1.260% | 19.608% | 23.81 | 24.11 | LA | $120,318.05 | $0.00 | $120,652.00 |
| 02/07/97 | 0.166% | 19.806% | 24.11 | 24.15 | LA | $120,517.66 | $0.00 | N/A |
| 02/10/97 | 0.497% | 20.401% | 24.15 | 24.27 | LA | $121,116.51 | $0.00 | $121,525.00 |
| 02/11/97 | 2.596% | 23.527% | 24.27 | 24.90 | LA | $226,856.25 | $100,000.00 | $227,275.00 |
| 02/12/97 | 0.201% | 23.775% | 24.90 | 24.95 | LA | $227,311.78 | $0.00 | N/A |
| 02/13/97 | 0.240% | 24.072% | 24.95 | 25.01 | LA | $227,858.42 | $0.00 | $228,304.00 |
| 02/14/97 | 1.279% | 25.659% | 12.51 | 12.67 | Gold | $230,772.68 | $0.00 | $231,224.00 |

Table 3.18: The trades over the real-time test period: January 28, 1997 to February 14, 1997. The first column is the date. The second column shows the daily return. The third column shows the inception to date return or cumulative return. The fourth and fifth columns are, respectively, the buy and sell prices. The sixth column indicates which mutual fund was traded (LA = Scudder Latin America Fund; PO = Scudder Pacific Opportunities Fund; Gold = Scudder Gold Fund; MM = Scudder Money Market Fund). The seventh column shows the estimated value of the account. The eighth column shows additions to the principal in the account. The ninth column shows the actual value of the account, as indicated by the brokerage firm. The actual capital in the account differs from the estimated capital for two reasons. First, the estimated amount assumes that the return of the money market fund is 0% when, in fact, it is slightly positive. Second, the estimated amount assumes that there are no costs associated with trading while the actual amount reflects these trading costs.

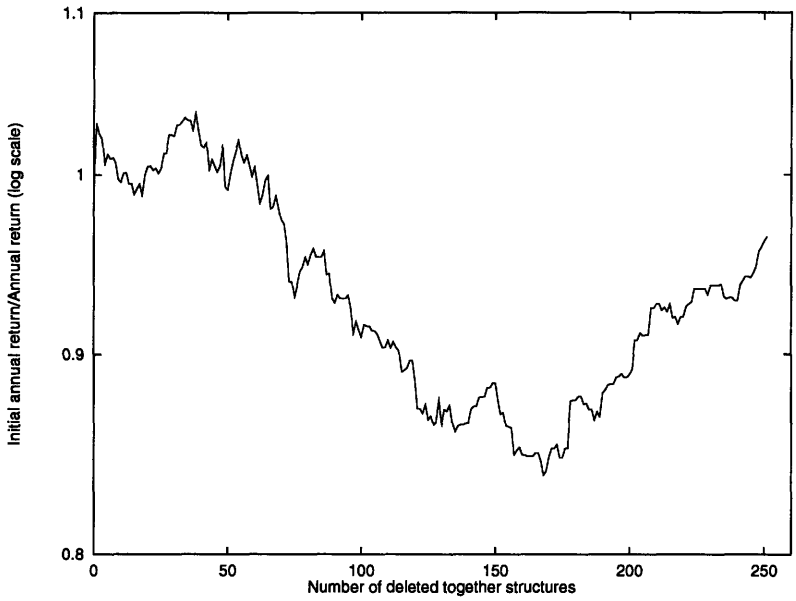


Figure 3-5: Ratio of annual return to initial annual return as a function of the number of deleted together structures. The 253 together structures that are the best covers on the out-of-sample data set are deleted sequentially, beginning with the one that covers the most days, and the annual return of the modified set of together structures is computed and divided by the return of the initial set of together structures. At the right most edge of the graph, all 253 initial together structures have been deleted and the annual return is 96.40% of the initial return. The minimum of the graph occurs after 168 together structures have been deleted. At that point the annual return is 83.82% of the initial annual return.

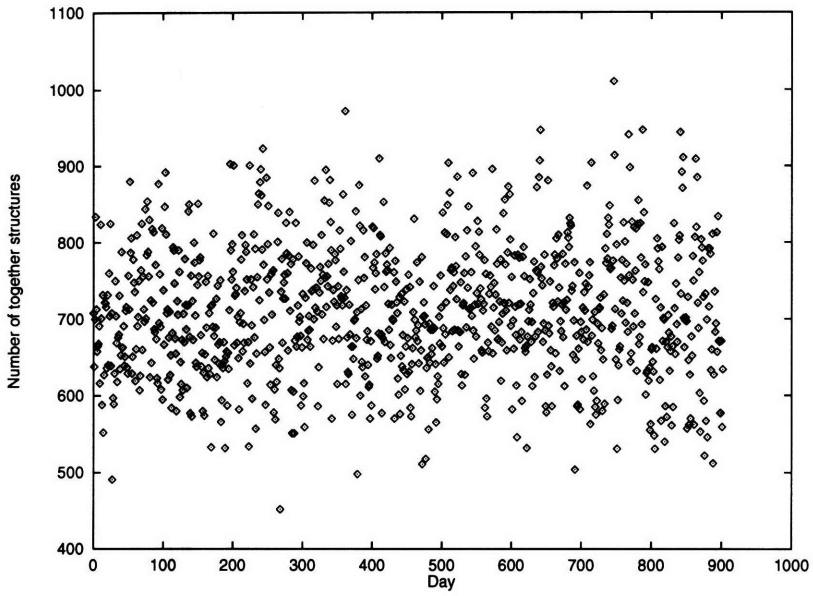


Figure 3-6: The number of together structures that cover each day in the out-of-sample data set. On average, 709.373 together structures cover each day in the out-of-sample data set. This redundant coverage explains why the performance of the system degrades gracefully when together structures are ablated.

average, 709.373 together structures (out of a total of 58,603; see Table 3.2) cover each day in the out-of-sample data set. Each of these together structures has been created by a beam search process that selects only the best together structures. Hence, although the best together structures may have been ablated, very good together structures are still available to take their place.

Figure 3-7 illustrates this phenomena for the most frequently occurring together structure in the out-of-sample data set. On eighteen of the twenty one days that the most frequently occurring together structure is the best cover, its replacement together structures (of which there are twelve; see Table 3.19) make the same prediction. On two of the remaining three days, the replacements actually outperform the original together structure.

Figure 3-8 shows, for each day on the out-of-sample data set, how many domain elements the best cover contains. None of the two domain element together structures is ever the highest accuracy covering together structure, so all of these structures could be deleted without impairing performance. Also, the four domain element together structure are over-represented. While less than 0.3% of the together structure

| Day | Replacement together structure |
|---|---|
| 302 | (1 0 7) (0 4 5) (0 6 4) |
| 339 | (1 0 10) (1 1 1) (1 5 1) |
| 340 | (1 0 10) (1 2 1) (1 3 1) |
| 352 | (1 0 10) (1 2 1) (2 9 8) |
| 510 | (0 0 8) (0 1 5) (0 8 1) |
| 511 | (1 0 10) (1 2 1) (1 3 1) |
| 518 | (1 0 10) (1 2 1) (1 6 1) |
| 521 | (1 0 10) (1 4 1) (1 5 1) |
| 536 | (1 0 10) (1 2 1) (1 6 1) |
| 548 | (0 0 9) (0 8 10) (2 2 3) |
| 549 | (1 0 10) (1 2 1) (1 3 1) |
| 559 | (1 0 10) (1 2 1) (2 9 8) |
| 560 | (1 0 10) (1 2 1) (1 3 1) |
| 561 | (1 0 10) (1 2 1) (1 3 1) |
| 562 | (1 0 10) (1 2 1) (1 3 1) |
| 563 | (1 0 10) (1 2 1) (1 3 1) |
| 619 | (1 0 7) (2 3 9) (2 4 9) |
| 625 | (1 0 10) (1 2 1) (2 5 5) |
| 654 | (0 0 8) (0 4 2) (2 7 8) |
| 724 | (1 0 8) (0 7 6) (1 7 4) |
| 815 | (1 0 10) (1 2 1) (1 6 1) |

Table 3.19: The together structures that replace (1 0 10) (1 1 1) (1 2 1). The Day column refers to the day in the out-of-sample data set on which the together structure listed in the second column is the best cover. The together structure that is the most frequent replacement is (1 0 10) (1 2 1) (1 3 1) which shares two domain elements with the original together structure.

```
(1 0 10) (1 1 1)  (1 2 1)
(0 0 10) (0 2 1)  (0 3 1)
(0 0 7)  (1 1 6)  (1 2 5)
(1 0 10) (0 3 1)  (1 5 1)
(1 0 10) (1 1 10) (1 2 10)
(0 0 10) (0 1 9)  (0 4 5)
(1 0 7)  (2 3 9)  (2 6 2)
(1 0 7)  (0 9 9)  (1 1 7)
(0 0 10) (1 3 5)  (1 5 4)
(0 0 10) (0 1 10) (0 3 10)
(0 0 8)  (0 4 2)  (2 7 8)
(0 0 7)  (0 2 7)  (0 4 9)
(1 0 10) (2 5 1)  (2 6 1)
(1 0 10) (1 2 10) (1 5 1)
(1 0 7)  (1 1 7)  (1 7 6)
(0 0 9)  (1 1 8)  (1 2 7)
(1 0 8)  (1 9 6)  (2 7 2)
(1 0 7)  (0 9 5)  (2 6 10)
(0 0 10) (1 1 10) (2 1 9)
(0 0 9)  (0 2 8)  (0 8 10)
(0 0 7)  (0 1 7)  (1 1 4)
(2 0 8)  (1 2 9)  (2 4 5)
(0 0 10) (0 1 9)  (2 6 8)
(0 0 8)  (0 4 2)  (1 9 5)
(0 0 7)  (0 5 6)  (0 9 7)
(1 0 10) (1 2 1)  (1 3 1)
(1 0 10) (1 1 10) (1 7 1) (2 7 1)
(1 0 10) (0 5 8)  (1 6 5)
(1 0 8)  (0 7 6)  (1 7 4)
(0 0 10) (0 2 10) (1 6 10)
(0 0 10) (0 1 1)  (0 2 1)
(0 0 8)  (1 7 4)  (2 3 7)
(2 0 8)  (0 7 7)  (1 3 10)
(1 0 10) (1 3 1)  (1 4 1)
(1 0 10) (1 2 1)  (1 6 1)
(1 0 10) (1 1 10) (1 4 1)
(1 0 10) (0 4 7)  (1 3 6)
```

Table 3.20: Together structures, ranked in order of the number of times that they are the best cover on the out-of-sample data set. Figure 3-7 shows how the performance on the out of sample data set changes when the first of these together structures is deleted. Figure 3-5 shows how performance degrades when each of these together structures is deleted.

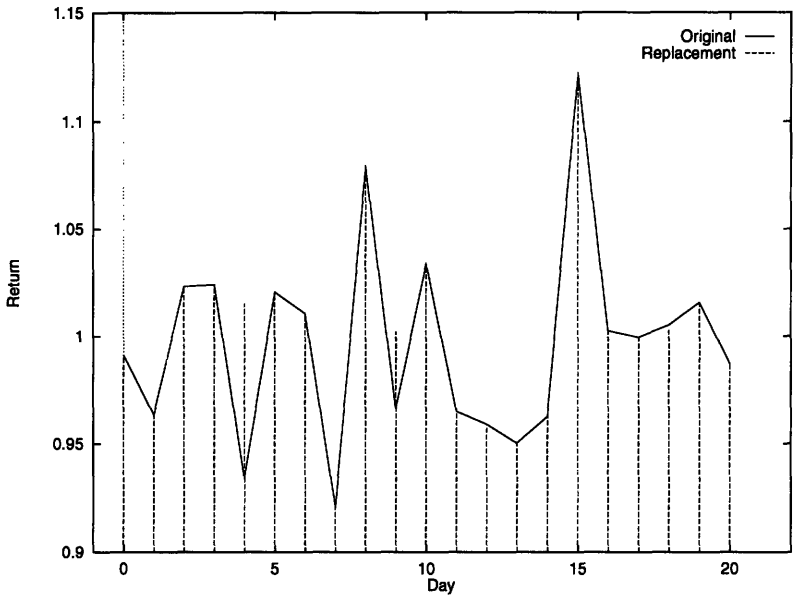


Figure 3-7: A comparison of the together structure that most frequently makes a prediction ("original") on the out-of-sample data set with its replacements ("replacement"). The straight line shows the return of the original together structure for each of the 21 days that it makes a prediction. The dotted line shows the performance of the together structure strategy when the original together structure is deleted. This figure illustrates the robustness of the together structure system.

database is composed of four domain element together structures, approximately 1.8% of the predictions are made by these together structures.

## 3.5 Together structures uncover local pockets of regularity

In this stock market application, the together structure representation captures regularities in the market that repeat over time, giving the system the opportunity to extract above average returns from the market. A short, but growing, real-time test of this strategy suggests that the returns achieved on the out-of-sample data can be realized.

Why does the relatively simple together structure strategy perform so well? Although I have no hard answers to this question, there are a few plausible reasons that suggest themselves.

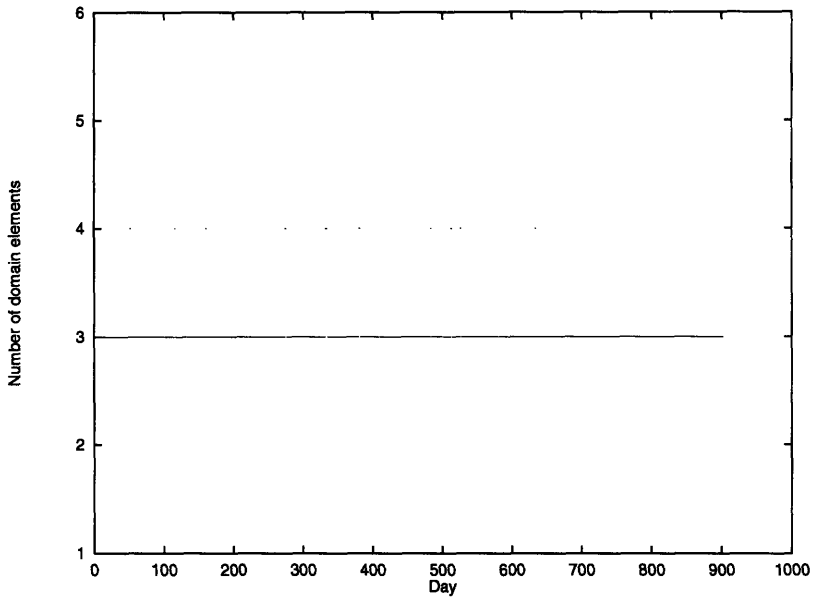


Figure 3-8: The number of domain elements in the best cover for each day of the out-of-sample data set. Note that although approximately 1% of the together structure memory consists of two domain element together structures (see Table 3.2) these together structures are always trumped by the three and four domain element together structures. Furthermore, the four domain element together structures are over-represented. Although they compose less than 0.3% of the together structures they are responsible for approximately 1.8% of the predictions. In short, there is a very clear hierarchy: the three domain element together structures are preferred to the two domain element together structures and the four domain element together structures are preferred to the three domain element together structures.

Part of the answer is certainly that mutual funds are relatively easier to trade simply because the transaction costs are so low compared to other financial instruments. So, while regularities may exist in other markets, they are not great enough to overcome transaction costs. However, this cannot be the full answer because even when transaction costs are added back into the returns very few market participants outperform a buy-and-hold strategy by a statistically significant margin [67].

Another answer is that together structures combine information from three different price series. This is in contrast to many technical trading strategies which are often based on a single price series. For example, *The Encyclopedia of Technical Market Indicators* [45] catalogs many technical indicators, none of which are functions of more than two price series. Indeed, in the field of devising technical analysis strategies, the idea of using more than one price series is new and unexplored [21].

Many tests of market efficiency (see, e.g., [66, 12, 50, 11, 10, 18, 19, 56, 32]) test whether one variable (such as yesterday's price change) influences future stock prices. If future stock prices are generated by an underlying model which has a large number of variables each of which has a small but non-negligible influence on the outcome, then such statistical tests will always fail to reject the null hypothesis that the market is efficient with respect to that variable. The together structure system, by building a large model composed of thousands of components each of which has a small but non-negligible influence on the performance, addresses, in part, this shortcoming.

A related, but more obscure, explanation is that, by binning daily returns, the together structure system operates at a lower level of abstraction than many other systems. To give but one example of how this might improve the performance, note that the second together structure in Table 3.5 indicates that a very poor performance in the Gold Fund is often followed by an excellent return in the Latin America Fund. That is to say, the returns are negatively correlated at the extremes. However, Table 3.13 shows that the Gold Fund and the Latin America Fund have a positive correlation of 0.40. So, an approach that computed the correlation of the two funds over the entire universe of daily returns would fail to find the local regularity that the together structure system uncovered. Table 3.21

| Lead (in days) | PO | LA |
|---|---|---|
| 1 | 0.075978 | 0.082169 |
| 2 | 0.042664 | 0.02098 |
| 3 | 0.057611 | 0.04398 |
| 4 | 0.040454 | 0.00912 |
| 5 | -0.00141 | 0.00990 |
| 6 | -0.03504 | -0.0342 |
| 7 | 0.006827 | -0.088 |
| 8 | -0.03581 | -0.0367 |
| 9 | -0.01165 | 0.03788 |

Table 3.21: Cross-correlation of the Gold Fund's daily return with the daily return of the Pacific Opportunities Fund (second column) and the Latin America Fund (third column).

| Lead (in days) | Gold | LA |
|---|---|---|
| 1 | 0.121869489 | 0.078176709 |
| 2 | 0.071808616 | -0.01430475 |
| 3 | 0.010141282 | -0.016510048 |
| 4 | 0.029733978 | -0.013278383 |
| 5 | 0.009987533 | 0.031928286 |
| 6 | -0.004112446 | 0.003682014 |
| 7 | 0.018793203 | -0.021098131 |
| 8 | 0.029294243 | 0.042329205 |
| 9 | 0.042713023 | 0.021894227 |

Table 3.22: Cross-correlation of the Pacific Opportunities Fund's daily return with the daily return of the Gold Fund (second column) and the Latin America Fund (third column).

| Lead (in days) | Gold | PO |
|---|---|---|
| 1 | 0.025300134 | 0.188988718 |
| 2 | 0.047805168 | 0.089690844 |
| 3 | -0.004054895 | 0.120566478 |
| 4 | 0.012045029 | 0.053048444 |
| 5 | 0.00531999 | -0.040458868 |
| 6 | -0.005779746 | -0.033212976 |
| 7 | 0.05799552 | 0.03307958 |
| 8 | 0.041022448 | 0.042330486 |
| 9 | 0.052882903 | 0.041669567 |

Table 3.23: Cross-correlation of the Latin America Fund's daily return with the daily return of the Gold Fund (second column) and the Pacific Opportunities Fund (third column).

A final answer to this question is that the together structure trading strategy is not simple at all. Although each together structure is simple, when the 58,000 together structures are combined into one strategy, the result is a complex, non-linear model of mutual fund prices.

# Chapter 4

# Application to chess

This section describes how a together structure based chess player learns how to play better chess by learning from a chess program that makes random moves. If you want to see the final board position of a game in which the player with access to the together structures defeats the random player, turn to Figure 4-6.

## 4.1 Chess is a difficult game with simple rules

Chess is played by two players on an 8x8 board as shown in Figure 4-1. Each player begins with sixteen pieces (eight pawns, two knights, two bishops, two rooks, one queen, and one king), and the players alternate moves until the king is attacked and cannot move into a square that is not attacked ("checkmate") or until the game is drawn.

## 4.2 The domain element is a piece on a board position

In the chess domain, domain elements could store information about a host of features, including:

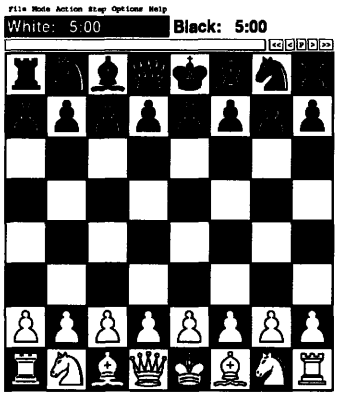


Figure 4-1: The starting state of the game of chess. The game is played on an 8x8 board. Each side has eight pawns, two rooks, two knights, two bishops, and one queen and king.

- Adjacency information. For example: "The white queen is aligned on the same diagonal as the white bishop."
- Move information. For example: "The white rook, black king, and white knight can all move to square d3 in one move."
- Spatial information. For example: "The white rook is on square e5."

For simplicity, I chose the third definition in this list.[1] With this definition the together structure system stores co-occurrence data about how often a board position with, say, a white queen on d3 and a black king on d8, leads to a win for white. So, each domain element has two components, one for the type of piece and one for the piece's position on the board. A blank square, or space, is considered to be a piece, as are the king, queen, rook, knight, bishop, and pawn.

[1]In addition to simplicity, this "spatial" domain element also has the property that it can be used to capture, in an admittedly indirect manner, adjacency information and move information.

## 4.3 Criteria of success

The goal of this application is to build a learning system for chess that can extract performance improving information from a database of chess games. Hence, the way to demonstrate success in this domain is to show that a chess playing program with access to together structures plays better than an identical chess playing program without access to together structures.

Informal experiments with standard chess programs, such as GNUCHESS and CRAFTY, suggested that important modifications would have to be made to them in order to incorporate together structures. The components of these programs have been tightly co-optimized so a change in one component forces changes in other components. Because changing multiple components of a standard program would defeat the purpose of using such a program, I chose instead to use a program that makes random chess moves.

So, the criteria of success for this application involves comparing a chess program that makes random moves to a chess program that instead of choosing a random move makes the move suggested by the together structures. If the chess player with access to the together structures outplays the random chess player, then this application of together structures will be considered a success.

Note that the together structures must be created by observing the random chess player play against itself. If the together structures were created by examining the games of chess masters, then the internal control provided by playing against the random chess player would be invalidated.

## 4.4 The learning procedure nests together structures

The source of data for the chess application is a database of games played by a random chess player against itself. A total of 416 games were played by the random player of which 26 were wins for white and 47 were wins for black. The rest of the games were

drawn.

The 73 games that were wins for white or black were used to build the together structures. Each game is processed by the together structure system after the completion of the game. After each move in each of these 73 games, each of the together structure that matches a board position is updated. For example, if on move 13 of a game in which white wins, the black king is positioned on d8 and the white queen is positioned on e5, then the together structure that stores information about these two domain elements ("black king on d8" and "white queen on e5") is updated by incrementing the slot that corresponds to the number of wins for black.

In the next pass through the games, each of the together structures created in the first pass is considered to be a domain element, and, as such, can be paired in a new together structure with another domain element. So, the second pass creates together structures with primitive domain element triples (when one of the together structures is paired with a primitive domain element) and quadruples (when two together structures are paired). Multiple passes are made through the games until there are no more together structures to extend. Because the data set is so small, every together structure which matches more than three or more board positions and which specifies the positions of ten or fewer pieces is extended. If the data set were larger, then some measure of the quality of the together structure in predicting a win for white or black would be used to decide which together structures to extend.

The end result of this process is a database of together structures, such as the ones shown in Figure 4-2. A total of 5,646 together structures are created from the database of 73 games.

```
S S    1 43  2 0
Q K   27 60  1 0
P N   32 36 11 0
P K   33 60  3 0
Q B   33 61  4 0
Q 521 33  x  1 0
```

Figure 4-2: Six together structures in the chess application. The first and second columns are either piece types or pointers to other together structures (S=space, P=white pawn, N=white bishop, B=white knight, R=white rook, Q=white queen, K=white king, number=reference to another together structure). The second two columns contain the piece location (the 64 squares of the chess board are numbered 0-63, x=reference to another together structure). The last two columns indicate the number of times that this position has led to a win for white and black, respectively. For example, the second together structure "Q K 27 60 1 0" can be read as: "There is one game in which white wins (column 5) when the white queen (column 1) is on square 27 (column 3) and the white king (column 2) is on square 60 (column 4)."

## 4.5 The program with access to together structures makes moves that favor winning board positions

Once the together structures are created from the games played by the random chess player against itself, a new chess program is created that is given access to these together structures (this player will be referred to as Tplayer from now on).

The Tplayer chooses a move by generating every possible move and then matching the together structures against the resulting board positions. The Tplayer selects the move that results in the board position that matches the most number of together structures that predict that the Tplayer will win. Approximately 2000 together structures match every board position. A small fraction of the together structures that match the board after white's rightmost pawn moves one square forward in the first move of the game is shown in Figure 4-3.

| Piece 1 | Piece 2 | X-1 | Y-1 | X-2 | Y-2 |
|---|---|---|---|---|---|
| S | B | 3 | 0 | 7 | 5 |
| S | N | 3 | 0 | 7 | 6 |
| S | R | 3 | 0 | 7 | 7 |
| S | S | 3 | 1 | 2 | 0 |
| S | S | 3 | 1 | 2 | 1 |
| S | S | 3 | 1 | 2 | 2 |
| P | S | 5 | 7 | 4 | 5 |
| P | S | 5 | 7 | 4 | 6 |
| P | S | 5 | 7 | 4 | 7 |
| P | S | 5 | 7 | 5 | 0 |
| P | S | 5 | 7 | 5 | 1 |
| P | S | 5 | 7 | 5 | 2 |
| P | P | 5 | 7 | 5 | 7 |
| P | P | 5 | 7 | 6 | 0 |
| P | P | 5 | 7 | 6 | 1 |
| P | P | 5 | 7 | 6 | 2 |
| P | P | 5 | 7 | 6 | 3 |
| N | S | 7 | 1 | 3 | 7 |
| N | S | 7 | 1 | 4 | 0 |
| N | S | 7 | 1 | 4 | 1 |
| N | S | 7 | 1 | 4 | 2 |
| N | S | 7 | 1 | 4 | 3 |
| N | S | 7 | 1 | 4 | 4 |
| N | S | 7 | 1 | 4 | 5 |
| N | S | 7 | 1 | 4 | 6 |
| N | S | 7 | 1 | 4 | 7 |
| N | S | 7 | 1 | 5 | 0 |
| N | S | 7 | 1 | 5 | 1 |
| N | S | 7 | 1 | 5 | 2 |
| N | S | 7 | 1 | 5 | 3 |

Figure 4-3: A small fraction of the two domain element together structures that match the board after white's rightmost pawn moves one square forward in the first move of the game . In the first and second columns $S$ stands for *space*, $P$ stands for *pawn*, $N$ stands for *knight*, $B$ stands for *bishop*, $R$ stands for *rook*, $Q$ stands for *Queen*, and $K$ stands for *King*. The final four columns show the x and y coordinates of the board position. The lower left hand corner of white's side of the board is (0,0).

## 4.6 Together structures improve chess playing ability

To test the quality of the together structures, the random chess program played against the Tplayer. The Tplayer chooses the move which leads to the board position which matches the greatest number of together structures that indicate a win for the Tplayer. For example, if the Tplayer has the white pieces and a together structure indicates that a board position has led to five wins for white and three for black, the score of that board position is incremented.

A total of 149 games have been played between the Tplayer and the random player. The Tplayer won two of the games and the random player won none. The rest of the games were drawn. The moves for the first game the TPlayer won are shown in Figure 4-4.

The games that resulted in draws all ended with an overwhelming advantage for the TPlayer. Using the standard scoring scheme[2] the average final score of the Tplayer was 21.9 versus one point for the random player. So the Tplayer had the equivalent of one queen, one rook, one bishop, one knight, and slightly under two pawns at the end of the drawn games while the average player had the equivalent of one pawn. One such drawn game is shown in Figure 4-5.

## 4.7 Discussion

An examination of the together structures and the Tplayer's "strategy" suggests that:

- The TPlayer plays very conservatively. It rarely moves its back row pieces during the beginning of the game.
- The TPlayer prefers placing the queen in the center of the board.

[2]Nine points for a queen, five points for a rook, three points for a bishop or a knight, and one point for a pawn

```
1. h3 e6
2. c3 f5
3. c4 h6
4. d4 Nc6
5. a4 Qg5
6. e4 Ne5
7. exf5 Rb8
8. fxe6 Nf6
9. e7 Qg4
10. exf8=B Qe2+
11. Qxe2 Rh7
12. Qd1 Rh8
13. Be7 b5
14. Bf8 Nd3+
15. Bxd3 Rb6
16. Bf1 d6
17. Be7 Rc6
18. Bf8 Nh5
19. Be7 Be6
20. Bf8 Nf6
21. Be7 d5
22. Bf8 Kxf8
23. Qb3 Ke7
24. Qxb5 Bd7
25. Qxd5 Be8
26. Qxc6 a6
27. Qa8 Ng4
28. Qa7 Kd7
29. Qa8 g5
30. Qa7 Ke6
31. Qa8 Kf5
32. Qa7 Bc6
33. Qa8 h5
34. Qa7 Ke4
35. Qa8 Ne3
36. Qa7 g4
37. Qa8 Nd1
38. Qa7 h4
39. Qa8 Rh5
40. Qa7 Bb7
41. Qa8 c6
42. Qa7 Rh6
43. Qa8 Rg6
44. Qa7 Rg5
45. Qa8 Nxf2
46. Qa7 g3
47. Qa8 Nxh1
48. Qa7 Rg6
49. Qa8 Rg4
50. Qa7 Rg6
51. Qa8 c5
52. Qa7 Re6
53. Qxc5 Bd5
54. Qxd5# 1-0
```

Figure 4-4: The moves of the first game won by the TPlayer. Note that the Tplayer repeats the same two moves between move 27 and 52.

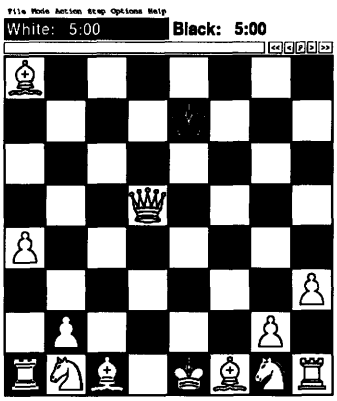


Figure 4-5: The final board position of a game between the Tplayer (white) and the random player (black) which illustrates the conservative play of the Tplayer. In this position, the Tplayer has yet to move most of its back row pieces from their initial starting positions.

The board position in Figure 4-5 illustrates the first point and the board position in Figure 4-6 illustrates the second.

Why does the together structure strategy learn these two chess concepts? In the training games (random player vs. random player) a piece that moves into the opponent's territory has a greater chance of being taken. So, the games which result in wins are, on average, the ones in which one of the players has kept its pieces outside of the enemy's territory. This is why the together structures encode a "conservative" strategy.

The second concept is learned from several games in which the queen plays an active role in mating from the center of the board. The final position of one such game is shown in Figure 4-7.

## 4.8 Related work

Beginning with Shannon's classic paper [62], an extraordinary amount of work has been done at the intersection of computer science and chess. Readers interested in a general introduction should consult Levy and Newborn [41], Newborn [48], and Berliner and Beal [7]. Here I concentrate on describing three lines of research that focus on learning and representation in chess.

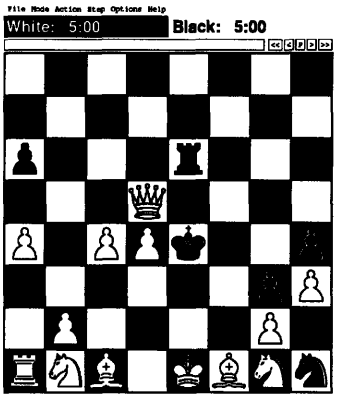


Figure 4-6: The final board position of game between the Tplayer (white) and the random player (black) which illustrates the Tplayer's preference for moving the queen into the middle of the board. In this position, the black king is checkmated.

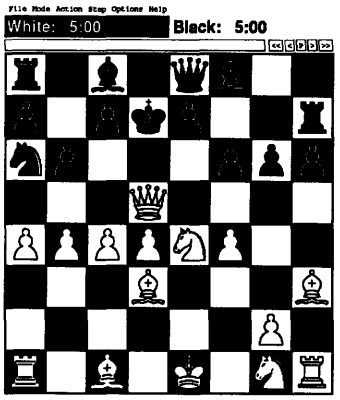


Figure 4-7: The final position of a game between the random player and itself. The centralized queen motif is repeated to good effect when the Tplayer plays against the random player, as illustrated in Figure 4-6.

Quinlan [55] develops a decision tree system for determining whether a small set of chess endgames are won or lost. Like the together structure system, the decision tree statically evaluates a position without performing a time consuming search. Quinlan investigates several different types of representations to determine which produce the smallest, fastest decision trees. The attributes he derives to represent the chessboard are all at a significantly higher level of abstraction than the piece position representation used by the together structure system. For example, one attribute is true exactly when the black King cannot move on the next move. Another attribute is: "white rook is safe from reprisal 2-ply by the black king if white king takes the knight or threatens to do so." The decision tree for one of the endgames runs approximately an order of magnitude faster than a specialized search routine.

Quinlan's program solves a problem that is slightly different from the one addressed by the together structure system. Quinlan is interested in creating perfect decision trees that always correctly classify a position into the won, lost, or draw categories. As such, his system takes as input an exhaustive list of all board positions, along with an indication of which category each position belongs to, and then creates a decision tree that compresses this data. Quinlan's system would not work in middle games because the set of possible positions is too large to enumerate. The together structure system avoids this complication by settling for creating heuristic rules that are generated from a very small subset of the total number of possible board positions.

Donninger [14] describes a graphical language for expressing chess knowledge which was developed with the goal of providing a tool that would enable chess experts to download their chess knowledge into a chess program without programming. The system allows the user to directly and graphically express three boolean relations: **not**, **or**, and **and**. This boolean language is used to define "patterns" which are then coupled with lists of good moves and weak moves called "advice." This pairing of patterns and advice is very similar to the one in the together structure system. The primary difference lies in how the patterns and advice are generated. In Donninger's system the information is added to the system by a human expert

while in the together structure system the pattern/advice pairs are learned from examples. Donninger, unfortunately, does not report any experiments which provide a quantitative account of how much the system improves after the graphical language is added.

For an extended period of time, Levinson [39, 26, 40] has pursued the goal of creating a "fully domain-independent single-agent and multi-agent heuristic search system." He has developed a chess playing program called MORPH, which like the together structure system, uses a very simple representation to capture information about which moves in a given position are likely to lead to winning outcomes. Its pattern language specifies attacking and defending relationships amongst pieces and square. Unfortunately, like the together structure system, Morph does not achieve a high level of playing ability. Levinson does not report any experiments which provide anything but anecdotal evidence that MORPH can learn how to play better chess.

# Chapter 5

# Contributions

In this chapter I summarize the contributions of this work, describe related research, and discuss how this work might be extended.

## 5.1 Summary of contributions

This research makes the following contributions:

First, the together system gives a proof by example that a weak learning algorithm coupled with a simple representation can capture performance improving information in three domains. By allowing the data to speak for itself, without imposing restrictive *a priori* assumptions, the together structure system has the opportunity to learn previously unrecognized regularities.

Second, the together structure system locates large sets of regularity capturing domain elements in a computationally efficient manner.

Third, the together structure system builds models with thousands of parameters. The performance of the system suggests that for some purposes this may be preferable to hand-crafting highly structured representations and *ad hoc* learning algorithms that focus on capturing only the few most salient features of a domain.

Fourth, each of the three domain applications of together structures makes a contribution to the domain. The stock market application described in Chapter 3 generated a trading strategy that has been implemented in real time. The information

retrieval application described in Chapter 2 produced a thesaurus of related items that has been incorporated into an information retrieval engine that is part of the START project. The chess application described in Chapter 4 suggests that brute force methods are not the only way to improve chess performance.

## 5.2 Related work

This section describes how marginal attribution, causal discovery programs, and concept learning systems are similar to and different from together structures. Related work for each of the applications is found in the corresponding chapters.

### 5.2.1 Marginal attribution

Drescher[15], motivated by Piagetian constructivism, invented an induction algorithm, called marginal attribution, which collects frequency information about pairs of actions and results.

The marginal attribution algorithm is similar to the together structure mechanism in at least two ways. First, within the scope of their respective representations, both do exhaustive data collection. The marginal attribution algorithm records information about every action and result pair, while the together mechanism collects information about every pair of domain elements. There is little bias or selection built into either algorithm. Second, both approaches store simple frequency information. Small changes in these frequencies do not qualitatively alter the behavior of the systems, thus both mechanisms avoid some of the brittleness associated with applying thresholds to statistical measures.

There are also important differences between the two approaches. Drescher has a strong idealogical stance that prohibits him from taking advantage of built-in knowledge or domain-specific biases, and, even more importantly, appears to have prevented him from providing a mechanism for incorporating this type of knowledge into his learning system which, although it may never have been used by him, may have been exploited by others. This work, on the other hand, is not based on such a theo-

retical bias and, in fact, most of the applications are expected to take advantage of domain-specific knowledge. One expected result of this difference between the two approaches is that together structures will be effective in solving real-world problems that are of practical interest, while Drescher's mechanism overwhelmed a Connection Machine CM2 computer while running in a simple world because it had to discover the simplest concepts from examples.

Second, the together mechanism is a passive observer which collects information from examples, while Drescher's system is implemented in an active observer which can, among other activities, focus on a particular action and result pair and thus collect information on it. The advantage of Drescher's approach is that this targeting, if accurate, can lead to faster improvements in performance. There are at least two disadvantages, however. First, it requires an additional control mechanism which allocates time between two qualitatively different behaviors. Second, in some domains this may not be possible. In the written text domain, for example, implementing the exploratory behavior would require having a teacher which can generate descriptions with particular properties.

### 5.2.2 Causality

Although the program described in this paper is not an implementation of a theory of causal inference, some of the problems encountered here are similar to those that have been partially addressed by research on causality.

In a wide-ranging thesis, Pazzani[51] describes a hybrid approach for inferring causal relationships. Pazzani clearly favors strong methods that provide substantial bias over the weak methods that are championed in this paper, and that is one of the important distinctions between our two approaches. Pazzani's approach has a short term performance advantage on specific problems and, indeed, the examples in the thesis are quite impressive, but over the long term and over a broad range of problems we would place our bets on weak methods.

Bozsahin and Findler[8] discuss a method for learning causal relations from episodic text which they implement in a program called NEXUS. This work is the most similar

to the one described here. There are several important differences, however. Bozsahin and Findler are interested in causal relations, not co-occurrences, and they use a set of handwritten causal heuristics in their program. As such, they make an intellectual commitment to causality and its underlying philosophical underpinnings, a commitment that we do not share. Their representations, both for the written text and for causal explanations, are much richer than together structures and were specifically created to facilitate reasoning about causal relationships. Their paper concludes with an excellent discussion of the shortcomings of their own work and of previous work on inferring causal relationships.

Glymour, Scheines, Spirtes, and Kelly[22] describe the Tetrad algorithm for uncovering causal relations in structured numerical data. In contrast, the work on together structures, as illustrated by the application to language understanding, places a much greater emphasis on domains in which domain elements cannot be adequately measured numerically and in which the number of domain elements can be very large. The book contains an excellent discussion of the philosophical and scientific issues that lie at the core of research on causal modeling and scientific induction.

Pearl[52], a pioneer in the field of reasoning under uncertainty, has proposed many methods for using known probabilities to compute unknown probabilities. Pearl's work presupposes that some probabilities are already known by the system. This work, in contrast, focuses on finding these initial probabilities. So, Pearl's systems could be layered on top of a together structure memory or used as a facilitating mechanism in certain tasks. Furthermore, one important distinction between the work of Pearl and the other work mentioned in this section is that Pearl appears to consider the problem of uncovering causal relationships to be a problem of deduction rather than induction.

### 5.2.3 Concept learning

The field of concept learning, best characterized by the work of Quinlan[54], concerns itself, in large part, with generating highly accurate predictive systems. These systems typically take as input an instance in the form of a set of attribute/value pairs and

produce as output a prediction.

Neural networks have been employed in a dizzyingly wide variety of applications including protein structure prediction [31, 36, 57], financial time series analysis [72, 34, 33, 37], chemical engineering [43, 24, 25], manufacturing [44, 4], and

These systems, like together structures, attempt to uncover regularities in streams of data. Unlike together structures, however, concept learning algorithms are generally created once the variable that is to be predicted has been determined. In addition, together structures, unlike most concept learning systems, explicitly store information about negative cases which facilitates reasoning about explanations.

### 5.2.4 Atkeson's memory-based system

Atkeson and his colleagues[5, 6, 46] have developed a memory-based control system that has been applied to many tasks, including devil stick juggling, billiards shot selection, and container filling. This system learns a forward model, a function which returns a behavior given a state and action, and an inverse model, a function which returns an action given a state and a behavior, by observing the action of a robot over repeated trials of the same task.

The system uses a feedback loop to select regions of the space to explore, thus allocating resources more efficiently than a static or open loop system. Time is not spent on learning regions of the space that are linear (because they are correctly approximated by a linear estimator from very few data points) and, instead, attention is focused on the non-linear regions of the space.

Atkeson notes that the system exhibits several desirable characteristics:

- The system automatically locates critical dimensions of the input space. Critical parts of the space are represented with high resolution (i.e., many data points) while less important regions of the space are represented at a coarser level of resolution.

- The locally weighted regression scheme automatically returns an uncertainty estimate that can be used to greatly reduce search time by focusing exploration

on regions of the space that have poor uncertainty estimates.

- Because the system stores exhaustive traces of previous experiments, it never repeats the same error.

- As a group, the local weighted linear regressions form a non-linear model of the space. This non-linear model is generated very quickly (compared to, say, neural networks [58, 27] and genetic algorithms[30, 23]) and it is not subject to local minima problems because it locates the model analytically without doing search.

- Using the model in real-time is computationally expensive because it requires fast access to a large memory of previous experiments.

The Atkeson approach and together structures are similar along several dimensions. First, they share a common architecture. Both create large memories that are composed of largely unmodified observations[1] and then probe this memory to determine which action to take. Atkeson's system creates a local model around the probe in real time while the together structure system pre-computes the desired action. Second, both systems are strongly data centered. *A priori* assumptions about the structure and distribution of the data are kept to a minimum and, hence, both systems fall under the rubric of weak learning systems.

There are several fundamental distinctions between Atkeson's approach and together structures. First, Atkeson's system generates new data points by experimenting. In the three together structure applications described in this thesis, the data is exogeneously provided and fixed. The together structure system, unlike Atkeson's system, contains no mechanism for intelligently exploring the space to locate examples that would improve performance. Second, the together structure system can work with a variety of different types of domain elements, while Atkeson's system operates only in real spaces. For domains in which there is no clear mapping onto

[1] Atkeson calls this "raw data."

the real numbers (such as chess and information retrieval) Atkeson's linear regression approach would have to be modified.

### 5.2.5 Data mining

Data mining [53] is a field which is dedicated to locating and describing regularities in large databases. It is a component of a larger process, often called knowledge discovery in databases, which includes, in addition to data mining, preprocessing and evaluating data (see Figure 5-1).

Agrawal, Imielinski, and Swami [1] describe three different types of database mining:

1. Classification. The classification task requires finding explicit tests that partition examples into disjoint classes. One of the best known such algorithms is Quinlan's C4 [54].

2. Associations. Learning associations involves learning rules of the form: $x \rightarrow y$ with probability $p$. An example of such a rule is: "80% of all clients who buy bread also buy butter."

3. Sequences. When a database contains ordered data, such as stock market prices, sequences of events can be extracted. An example of such a rule is: "Civil wars conflicts are preceded by civil conflict 80% of the time."

Important applications have been developed, implemented, and employed in all three categories.

Among categorization applications, one stand out is the Financial Crimes Enforcement Network [61] which examines bank transactions and alerts users to transactions that may involve laundering.

In the association category, Mannila, Toivonen, and Verkamo [42] describe an application on a telephone company fault management database which has 30,000 examples and 210 attributes per example. This database is significant larger than those found in the UCI Machine Learning Depository, a popular database of databases that is often used to benchmark concept learning algorithms.

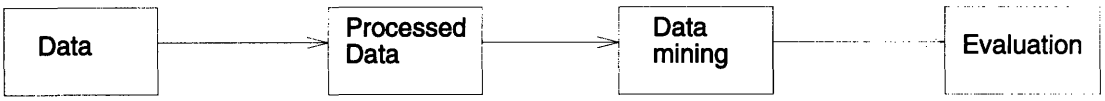


Figure 5-1: Steps in knowledge discovery. This thesis focuses on data mining, one of the critical steps in the knowledge discovery process.

In the sequence category, LBS Capital Management, an Inc. 500 firm, has developed a hybrid algorithm that combines expert systems, genetic algorithms and neural networks to manage a portfolio that exceeds half a billion dollars [28]. Since inception, their system has outperformed the broad stock market on a risk-adjusted basis. The application described in Chapter 3 of this thesis is also an example of such an application.

## 5.3 Potential next steps

In this section, I describe modifications that could be made to improve the system's performance.

### 5.3.1 Improving the together structure selection function

In each of the three applications described in this thesis, a critical step of the together structure procedure involves choosing which together structure to apply. In the stock market domain, described in Chapter 3, the together structure with the highest accuracy selects the fund to invest in. In the chess domain, described in Chapter 4, each together structure is given equal weight

A better selection function would combine aspects of both of these selection functions. This selection function would weigh better together structures more heavily, as in the stock market application, and would give some weight to all of the together structures that match the current example, as in the chess application.

To reduce this question to mathematical form, the selection procedure should be able to estimate P($W$—$A$&$B$) given P($W$—$A$) and P($W$—$B$). $A$ and $B$ are together structures that match the current example and $W$ is one of two possible outcomes for

the current example. Imagine, for example, that there are two outcomes for a chess game, a win for white or a loss for white, and that two together structures match the current board position. P($W$—$A$) and P($W$—$B$) are given by the frequencies that the together structures carry.

The key issue in generating a formula for P($W$—$A\&B$) is to decide which assumption to make. Note that, provided that P($W$—$A$) and P($W$—$B$) are neither 0 nor 1, this probability can range between 0 and 1. For example, suppose that $W$ is "The Cowboys win the football game" and $A$ is "The Cowboys outscore their opponents in the first half" and $B$ is "The Cowboys outscore their opponents in the second half" then P($W$—$A\&B$)=1 no matter what P($W$—$A$) and P($W$—$B$) are. Similarly, if $A$ is "The opposing team outscores the Cowboys in the first half" and $B$ is "The opposing team outscores the Cowboys in the second half" then P($W$—$A\&B$)=0 no matter what P($W$—$A$) and P($W$—$B$) are.

Fortunately, assuming that P($W\&A$) and that P($W\&B$) are independent, then:

$$P(W|A\&B) = ab/(ab + (1-a)(1-b)) \tag{5.1}$$

where $a$ =P($W$—$A$) and $b$ =P($W$—$b$).

A plot of this function is shown in Figure 5-2.

I expect that this selection function would lead to the greatest improvement in the chess domain. In the stock market domain, the best selection function would satisfy some economic criteria, such as maximizing expected returns.

Even this selection function, however, is not the best that can be imagined. For example, suppose that because of poor sampling the together structure estimates that P($W$—$A$)=0 when in fact P($W$—$A$)=0.0001. Then, whenever $A$ matches the current example, P($W$)=0 if equation 5.1 is applied, no matter how many other together structures that predict that $W$ is very likely match the current situation. One way to address this problem would be to assume some prior on the conditional probabilities and combine them with the probabilities given by the together structures' frequencies.

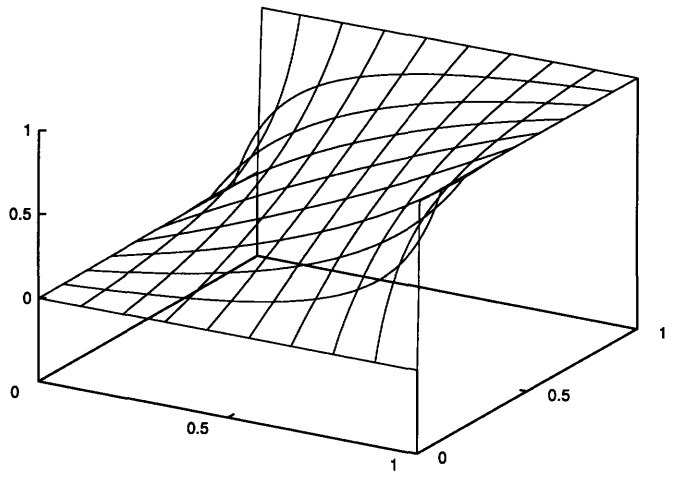


Figure 5-2: The function $f(a,b) = \frac{ab}{(ab+(1-a)(1-b))}$. The $x$ axis is $a$, the $y$ axis is $b$, and the $z$ axis is $f$. This function describes what is the probability that an event will occur, given that there are two together structures that predict that the event will occur. One together structure predicts that the event will occur with probability $a$ and the other predicts that the event will occur with probability $b$.

### 5.3.2 Soft matches might improve performance

Together structures are currently updated only if they match an example exactly. A line of work in articial intelligence, typified by the fuzzy logic field[71], suggests that soft matches might improve performance.

The inability to perform soft matches is particularly glaring in the stock market application, described in Chapter 3. In this application, stock market returns are divided into bins. A stock market return which is very close to the edge of a bin is counted as being a full member of that bin. However, because the endpoints of the bins are determined empirically, a small change in the data set might cause significant changes in which regularities are uncovered by the together structure system.

One way to address this issue would be to assign a graded membership score to several bins that would be inversely proportional to a distance metric so that bins that were closer to the stock market return would have a higher membership score than those that are far away. Note that this would not require changing the frequency counts in the together structures, although their interpretation would have to change.

### 5.3.3 Creating a category hierarchy could improve inferences

The together structure representation, as described in this thesis, is flat: there are no hierarchies of objects. There is a single representational unit, the together structure, which contains recursively nested domain elements.

Creating hierarchies, or groups of together structures that belong to a single class, might improve inference capabilities. For example, suppose that the system had previously determined, by some method, that two together structures worked well in the same situations, say in predicting the result of chess end games with one bishop. Then, if evidence was collected that one together structure also performed well in another situation, say chess end games with two bishops, then it could be inferred that the other together structure would also work well in that situation.

How might these categories be automatically created? One idea is to implement the following heuristic: together structures that work well in many of the same situations can be considered to be members of the same category. This idea is illustrated in Figure 5-3.

How might this idea be implemented in practice? Consider the set of sentences shown in Figure 5-4. Suppose that this sentences were typed into a together structure system similar to the one described in Chapter 2 which parses sentences into subject-relation-object triples and then stores co-occurrence information about co-occurrences of these triples in together structures.

Now suppose that the system is given as input the three segments in Figure 5-5. These three segments are identical to the first three segments of Figure 5-4 except that "John" has been replaced by "Sally". Because "John" and "Sally" have identical relations to multiple objects, they could be grouped together in a single category.

If this category formation took place then in answer to the question: "Sally broke her leg. She goes to the hospital. Does she have a high fever?" the system could conclude that Sally does not have a high fever because in the fourth sentence of Figure 5-4 John, in an identical situation, did not have a high fever.

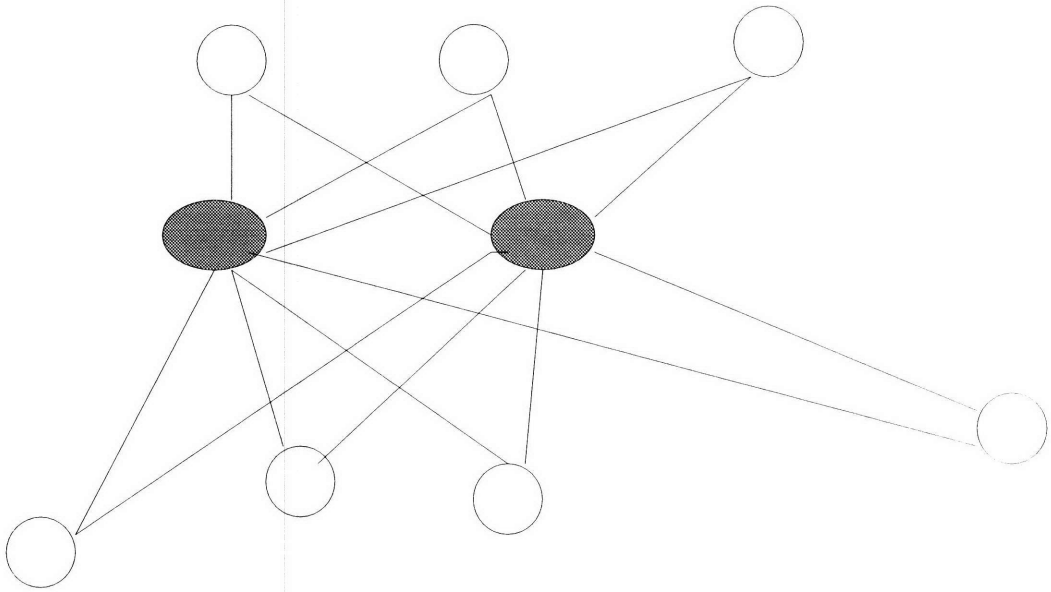

Figure 5-3: Two objects, shown in dark circles, which have identical relations to the same objects can be considered to be members of the same category. By applying this heuristic across the together structure memory, an IS-A hierarchy can be automatically generated.

- John is sick. He goes to the hospital.
- John is sick. He broke his leg. He goes to the hospital.
- John is sick. He has a high temperature. He goes to the hospital.
- John broke his leg. He goes to the hospital. He does not have a high temperature.

Figure 5-4: A set of sentences that, when processed by the together structure system, establish a set of relationships for the subject "John".

1. Sally is sick. She goes to the hospital.
2. Sally is sick. She broke his leg. She goes to the hospital.
3. Sally is sick. She has a high temperature. She goes to the hospital.

Figure 5-5: Three sentences that are identical to the first three sentences in Figure 5-4 except that "Sally" has been replaced by "John." Because "Sally" and "John" have identical relationships to multiple objects, they could be grouped to form a category.

In practice, the situation would not be as simple as the one given in this example. For example, subjects would not have identical relations to multiple objects. So some form of heuristic cutoff (e.g., "If two subjects have identical relations to no fewer than half of the objects they have in common, then form a category") would have to be implemented.

# Bibliography


[1] Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Database mining: A performance perspective. *IEEE Transactions on Knowledge and Data Engineering*, 5(1):914–925, 1993.

[2] Rakesh Agrawal and Ramakirshnan Skrikant. Fast algorithms for mining association rules. In *Proceedings of the 20th VLDB COnference*, 1994.

[3] D.W. Aha, D. Kibler, and M.A. Albert. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.

[4] K.S. Ali, D.S. Ali, and A.L. Ali. A neural networks approach to flexible manufacturing systems. In *Proceedings of the 1988 Summer COmputer Simulation Conference*, San Diego,CA, 1988.

[5] C.G. Atkeson. Using local models to control movement. In *Proceedings of Neural Information Processing Systems Conference*, 1989.

[6] C.G. Atkeson. *Advances in Neural Information Processing Systems 6*, chapter Using Local Trajectory Optimizers to Speed up Global Optimization in Dynamic Programming. Morgan Kaufmann, 1994.

[7] Hans J. Berliner and Don F. Beal. Introduction to the special issue on computer chess. *Artificial Intelligence*, 43(1):1–5, 1990.

[8] H. C. Bozsahin and N. V. Findler. Memory-based hypothesis formation: Heuristic learning of commonsense causal relations from text. *Cognitive Science*, 16(4):431–454, 1992.

[9] P.R. Cohen and A.E. Howe. How evaluation guides AI research. *AI Magazine*, 9(4):35–43, 1988.

[10] Paul H. Cootner and Sidney S. Alexander. Stock prices; random versus systematic changes. *Industrial Management Review*, 3:24–25, 1921.

[11] Paul H. Cootner and Sidney S. Alexander. Price movements in speculative markets: Trends or random walks. *Industrial Management Review*, 2:7–26, 1961.

[12] Alfred Cowles and Herbert E. Jones. Some a posteriori probabilities in stock market action. *Econometrica*, 5:294, 1937.

[13] C. Davidson. Science unravels the structure of the financial markets. *Thesis*, 1995.

[14] C. Donninger. CHE: A graphical language for expressing chess knowledge. *ICCA Journal*, 19(4):234–241, December 1996.

[15] G. Drescher. *Made-Up Minds: A Constructivist Approach to Artificial Intelligence.* MIT Press, Cambridge, MA, 1991.

[16] S. T. Dumais, G. W. Furnas, T. K. Landauer, S. Deerwester, and R. Harshman. Using latent semantic analysis to improve access to textual information. In *Proceedings of CHI '88*, pages 281–285, 1988.

[17] Federation International Des Eches. *The Official Laws of Chess and Other Fide Regulations.* BT Batsford Ltd, 1989.

[18] Eugene Fama. The behavior of stock market prices. *Journal of Business*, 38, January 1965.

[19] Eugene Fama. Efficient capital markets: A review of theory and empirical work. *Journal of Finance*, 25:383–423, 1970.

[20] S.I. Gallant, W.R. Caid, J. Carleton, R. Hecht-Nielson, K.P. Qing, and D. Sudbeck. Hnc's *matchplus* system. In *TREC-1*, pages 107–111, 1993.

[21] Michael E. Gayed. *Intermarket analysis and investing: integrating economic, fundamental, and technical trends.* New York Institute of Finance, New York, 1990.

[22] C. Glymour, R. Scheines, P. Spirtes, and K. Kelly. *Discovering Causal Structure: Artificial Intelligence, Philosophy of Science, and Statistical Modeliing.* Academic Press, Inc., 1987.

[23] D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, MA, 1989.

[24] R. Goodacre, A. N. Edmonds, and D. B. Kell. Quantitative analysis of the pyrolysis–mass spectra of complex mixtures using artificial neural networks: application to casamino acids in glycogen. *Journal of Analytical and Applied Pyrolysis*, 26:93–114, 1993.

[25] R. Goodacre, D. B. Kell, and G. Bianchi. Rapid asessment of the adulteration of virgin olive oils by other seed oils using pyrolysis mass spectrometry and artificial neural networks. *Journal of the Science of Food and Agriculture*, 53(3):297–307, 1993.

[26] J. Gould and R. Levinson. Experience-based adaptive search. In R. Michalski and G. Tecuci, editors, *Machine Learning: A Multi-Strategy Approach*, volume 4, pages 579–604. Morgan Kaufmann, 1994.

[27] S. Grossberg. *Neural Networks and Natural Intelligence.* MIT Press, Cambridge, MA, 1988.

[28] J. Hall, G. Mani, and D. Barr. Applying computational intelligence to the investment process. In *Proceedings of CIFER-96: Computational Intelligence in Financial Engineering*, Washington, DC, 1996. IEEE Computer Society.

[29] Donna Harman, Dennis Benson, Larry Fitzpatrick, Rand Huntzinger, and Charles Goldstein. IRX: An information retrieval system for experimentation and user applications. *SIGIR Forum*, 22(3-4):2–10, 1988.

[30] J. H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, 1975.

[31] L.H. Holley and M. Karplus. Protein secondary structure prediction with a neural network. *Proceedings of the National Academy of Science*, 86:152–156, 1989.

[32] Robert A. Huagen. *The New Finance: The Case Against Efficient Markets.* Prentice Hall, 1995.

[33] J. Hutchinson, A. Lo, and T. Poggio. A nonparameteric approach to pricing and hedging derivative securities via learning networks. *Journal of Finance*, 49:851–889, 1994.

[34] James M. Hutchinson. *A Radial Basis Function Approach to Financial Time Series Analysis.* PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 1994.

[35] B. Katz. Using English for indexing and retrieving. In P. H. Winston and S. A. Shellard, editors, *Artificial Intelligence at MIT: Expanding Frontiers*, volume 1. MIT Press, Cambridge, MA, 1990.

[36] D.G. Kneller, F.E. Cohen, and R. Langridge. Improvements in protein secondary structure prediction by an enhanced neural network. *Journal of Molecular Biology*, 214:171–182, 1990.

[37] Stephen R. Lawrence. Rule extraction for financial prediction. *Neural Information Processing Systems*, 9, 1996.

[38] Edward E. Leamer. Let's take the con out of econometrics. *American Economic Review*, 73:31–43, 1983.

[39] R. Levinson and R. Snyder. Adaptive pattern oriented chess. In *Proceedings of AAAI-91*, pages 601–605, 1991.

[40] Robert Levinson. General game-playing and reinforcement learning. Technical Report UCSC-CRL-950-06, University of California, Santa Cruz, 1995.

[41] David Levy and Monroe Newborn. *All about Chess and Computers: Containing the Complete WOrks, Chess and Computers (Computer Chess Series).* Computer Science Press, 1983.

[42] Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. Efficient algorithms for discovering association rules. In *Knowledge Discovery in Databases, Papers from the 1994 AAAI Workshop (KDD '94)*, pages 181–192, Seattle, Washington, July 1994.

[43] T. J. McAvoy, N. S. Wang, S. Naidu, N. Bhat, J. Gunter, and M. Simmons. Interpreting biosensor data via backpropagation. In *Proceeding of the 1989 International Joint Conference on Neural Networks (IJCNN '89)*, volume 1, pages 227–233, 1989.

[44] C. H. Messom, C. J. Hinde, A. A. West, and D. J. Williams. Designing neural networks for manufacturing process-control systems. In *Proceedings of the 1992 IEEE International Symposium on Intelligent Control*, Glasgow, Scotland, 1992.

[45] T. Meyers and R. Colby. *The Encyclopedia of Technical Market Indicators.* Dow Jones-Irwin, Homewood, IL, 1988.

[46] A.W. Moore, C.G. Atkeson, and S. A. Schaal. Memory-based learning for control. Technical Report CMU-RI-TR-95-18, Carnegie Mellon University, 1995.

[47] Victor Neiderhoffer. *The Education of a Speculator.* John Wiley, New York, 1996.

[48] Monroe Newborn. Computer chess: Ten years of significant progress. *Advances in Computers*, 29:197–250, 1989.

[49] Greg Newton. Top ten hedge funds and funds of funds ranked by return and risk-adjusted return. *HedgeMAR*, pages 34–35, January 1997.

[50] M.F.M. Osborne. Brownian motion in the stock market. *Operations Research*, 7(2), March-April 1959.

[51] M. J. Pazzani. *Learning causal relationships: an integration of empirical and explanation-based learning methods.* PhD thesis, UCLA, 1988.

[52] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1988.

[53] G. Piatetsky-Shapiro. Knowledge discovery in real databases: A report on the ijcai-89 workshop. *AI Magazine*, 11(5):68–70, 1996.

[54] J. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1993.

[55] J.R. Quinlan. Learning efficient classification procedures and their application to chess end games. In R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach.* Morgan Kaufmann, 1983.

[56] Paul A. Reeder. The efficacy of fundamental and technical ranking systems: tests of the efficient market hypothesis. Master's thesis, MIT, 1985.

[57] B. Rost and C. Sander. Prediction of protein secondary structure at better than 70% accuracy. *Journal of Molecular Biology*, 232:584–599, 1993.

[58] D. E. Rumelhart and J. L. McClelland, editors. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition.* MIT Press, Cambridge, MA, 1986.

[59] G. Salton, J. Allan, and C. Buckley. Automatic structuring and retreival of large text files. *Communications of the ACM*, 37(2):97–108, February 1994.

[60] C. Scott and R. Smith. *Innovative Applications of Artificial Intelligence 3.* AAAI Press, Menlo Park, CA, 1992.

[61] T Senator, H. G. Goldberg, J. Wooton, M.A. Cottini, A.F. Umarkhan, C. D. Klinger, W. M. Llamas, M. P. Marrone, and R. W. H. Wong. The financial crimes enformement network ai system (fais): Identifying potential money laundering from reports of large cash transactions. *AI Magazine*, 16(4):21–39, 1995.

[62] C. Shannon. Programming a computer for playing chess. *Philosphical Magazine*, 41(314):256–275, March 1950.

[63] H. Shrobe. The Innovative Applications of Artificial Intelligence Conference: Past and future. *AI Magazine*, 17(4):15–20, 1996.

[64] Ramakirshnan Skrikant and Rakesh Agrawal. Mining generalized association rules. In *Proceedings of the 21st VLDB COnference*, 1995.

[65] Lynn Andrea Stein. The Innovative Applications of Artificial Intelligence Conference: Past and future. *AI Magazine*, 17(4):77–83, 1996.

[66] Frank Taussig. Is market price determinate? *Quarterly Journal of Economics*, 25:394–411, May 1921.

[67] Richard J. Teweles and Frank Joseph Jones. *The futures game: Who wins? Who loses? Why?* McGraw-Hill, New York, 1987.

[68] Howard Robert Turtle. *Inference Networks for Document Retrieval.* PhD thesis, University of Massachusetts, 1991.

[69] Patrick Henry Winston. *Artificial Intelligence.* Addison-Wesley Publishing Company, Reading, MA, third edition, 1992.

[70] D. Yuret and M. de la Maza. A genetic algorithm system for predicting the OEX. *Technical Analysis of Stocks & Commodities*, pages 58–64, June 1994.

[71] L. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[72] X. Zhang and J. Hutchinson. Simple algorithms on fast machines: Practical issues in nonlinear time series prediction. In A. S. Weigend and N. A. Gershenfeld,

editors, *Time Series Prediction: Forecasting the Future and Understanding the Past*, pages 219–241, Reading, MA, 1993. Addison-Wesley.