Massively Parallel AI*

David L. Waltz

Thinking Machines Corporation
245 First Street
Cambridge, MA 02142-1214

Brandeis University
Computer Science Department
Waltham, MA 02254

Abstract

Most AI researchers would, I believe, agree that truly intelligent machines (i.e. machines on a par with humans) will require at least four orders of magnitude more power and memory than are available on any machine today [Schwartz 1988, Waltz 1988]. There is now widespread agreement in the supercomputing community that by the year 2000 all supercomputers (defined as the most powerful machines available at a given time) will be massively parallel [Fox 1990]. Yet relatively little thought has been given in AI as to how to utilize such machines. With few exceptions, AI's attention has been limited to workstations, minicomputers and PCs. Today's massively parallel machines present AI with a golden opportunity to make an impact, especially in the world of commercial applications. The most striking near-term opportunity is in the marriage of research on very large databases with case-based and memory-based AI. Moreover, such applications are steps on a path that can lead eventually to a class of truly intelligent systems.

Economic Inevitability of Massive Parallelism

The performance of serial computers is limited by the "von Neumann bottleneck," (the serial path used to move instructions and data between memory and the CPU) and by I/O limitations. Over the next ten years the use of fast technologies (ECL, GaAs, etc.) and further miniaturization might gain a performance factor of five, cleverer caching and instruction prefetch a factor of two, and the use of multiple functional units yet another factor of four, bringing the fastest uniprocessors (now at less than one GFlops) to perhaps 40 GFlops. Compiler technologies could allow as many as 16 such processors to be ganged together, yielding perhaps as much as 640 GFlops in total. The price would be high: very dense chips of exotic materials packed close together present daunting cooling and packaging problems. An extrapolation of current trends (supercomputers cost about $100,000/MFlops in 1977, and about $8000/MFlops in 1990) suggests that such a machine would cost about $500 million in the year 2000 (about $800/MFlops).

In contrast, massively parallel machines (e.g. the Connection Machine(R) CM-2) use PC/workstation technologies, and boast per-computational-unit costs similar to those of these small machines: on the order of $1000/MFlops in 1990. The largest massively parallel machines already exceed the power of serial supercomputers: the 65,536 processor CM-2 is realistically capable of speeds on the order of 5 GFlops (with a 28 GFlops peak). DARPA has targeted a massively parallel TeraOps (one trillion operations/second) machine by 1995, and the cost is expected to be less than $100 million ($100/MFlops). By the year 2000 it should be possible to build massively parallel TeraOps machines for $10 million ($10/MFlops - two orders of magnitude cheaper than could be done with serial technologies).

A Brief History of Massively Parallel AI

The Connection Machine system was originally designed to handle AI tasks, such as NETL-like marker-passing over semantic networks, and low-level computer vision [Fahlman 1979, Hillis 1985]. *Lisp, an extension of CommonLisp, was the first high-level language for the CM and its first front-end computer was a Symbolics 3600. A number of the early CMs were purchased by AI labs following its introduction in 1986, and some early work was done along the lines envisioned by Hillis: for example, CIS, a marker-passing parallel expert system, with one (instantiated) rule per processor [Blelloch 1986], and a system for computing stereo disparity from a pair of images, using the Marr-Poggio method [Marr and Poggio 1976, Drumheller 1986]. (See [Waltz & Stanfill 1988] for a summary of several early AI projects.) However, it is clear in retrospect that there was not much of an AI market for $50,000 Lisp machines, let alone for $1 million (and up) Connection Machine systems performing these sorts of applications. There still is not a market for very large expert systems, in part because it is difficult to build a very large expert system (Blelloch's CIS system contained 100,000 artificially generated rules on a CM-1 with about 8 MBytes of memory; a current

*This work was supported in part by the Defense Advanced Research Projects Agency, administered by the U.S. Air Force Office of Scientific Research under contract #F49620-88-C-0058 and is also funded in part by the United States Bureau of the Census.

CM-2 with 8 GBytes of memory could store 100 million rules!). Likewise, there was not (and still is not) a market for high volume low-level image processing (although the new Landsats may eventually create one).

Fortunately for Thinking Machines, the CM-2, with its floating point option (using up to 2000 Weitek chips), proved to be an excellent match for a wide range of scientific problems: finite element models, operations on large matrices, fluid flow and aerodynamics models, n-body problems (e.g. galactic collisions), interactive scientific visualization, models using cellular automata, seismic data processing, signal processing, etc. Most of Thinking Machines' sales growth -- 50% per year over the past four years -- has been fueled by sales for these types of applications.

But the AI dream is still valid. Recent research using novel AI paradigms on real customer problems and real customer databases have identified some great opportunities.

The Large-Scale Commercial AI Opportunity

Data is being generated faster than it can be digested. Very large (>100 GByte) databases are becoming common. Such databases contain valuable information on customer credit and buying behavior; many forms of text: public text, including books, newspapers and news wires, financial journals, technical industry publications, scientific journals, and annual reports; and private text, such as studies, memos, manuals, proposals, documentation; visually oriented materials, including maps, schematics, plans and blueprints, as well as fast-expanding video archives; and much more.

Until recently mainframe computers, with their extensive existing software libraries, have been the only choice for those who own and/or wish to mine such databases. But mainframes are pitifully slow and at the same time very expensive, and the software available for database tasks has been quite primitive because answering even primitive questions takes too long. Massively parallel machines offer large storage capacity[1] and high I/O rates, using parallel disk arrays and multiple wide-word I/O channels. The total CM-2 I/O capacity is currently 200 MBytes/second, limited by the disk rates.[2] This means that an entire 100 GByte database can be streamed into the CM-2 in 500 seconds. We estimate based on preliminary but realistic tests, that we can achieve a speedups of at least an order of magnitude over identical data selection operations on a mainframe on a CM-2 costing only a fraction of the price. [Stanfill, forthcoming].

This kind of performance opens the possibility for changing the fundamental nature of an organization's use of a database -- from a weekly to a daily batch run, or from batch to interactive -- and also opens the opportunity for applying dramatically more complex and intelligent processing to each database item than can otherwise be imagined today. In the next several sections, I outline the algorithms and performance for several commercially important applications.

Text-based Intelligent Systems

For most of its history, AI has been concerned with "toy problems." Scaling up presents difficulties: At the two extremes of the spectrum, one can hand-code (as in CYC [Lenat et al. 1986]), or one can use methods to automatically build NLP systems. To date there have been very few practical applications of natural language processing, and this fact has dampened the enthusiasm of funding agencies and companies that support research in this area. Fortunately, there are signs that this situation can be improved by strategically merging AI/NLP and Information Retrieval (IR) technologies. This offers novel opportunities both for learning research, and for building systems that have immediate practical value.

A series of experiments and discoveries led researchers at Thinking Machines, most notably Craig Stanfill and Brewster Kahle, to devise a document retrieval system that works in parallel on the Connection Machine [Stanfill & Kahle 1986; Stanfill 1988]. The resulting system, marketed as DowQuest(R) by Dow Jones has been in commercial use since January 1989. DowQuest provides a high quality search through a clever interface that can be used effectively by a computer-naive person after only about 5 minutes of training. The basic idea is this: A database of documents (e.g. news articles, abstracts, books, etc.) is distributed to each of the 65,536 processors of a CM-2 (if documents are 2K bytes long, each processor equipped with large memory can hold about 256 compressed documents or 16 million total!). The user types a few words (a question, description, or list of terms will do) and a carriage return; the terms are broadcast to all the processors in parallel along with a numerical "weight" indicating the importance of each term.[3] The search portion of this operation is analogous to the following situation: imagine a stadium with 65,536 people, each with one document and a pocket

[1] A single current Connection Machine CM-2 can be equipped with up to eight I/O channels, each of which can have 16 drops, allowing 15 DataVault disk arrays of up to 80 GBytes each, for a total storage capacity of 8x15x80 GBytes = 9,600 GBytes!

[2] The CM-2 itself can handle more than 400 MBytes/second.

[3] Term weights are assigned automatically by a program that pre-processes the text and updates the database. The number of occurrences of each term is saved, and weights computed proportional to the negative log of the probability of occurrences of each term.

calculator; an announcer reads each search term followed by a number (representing the importance of a term (the rarer a term, the higher its weight) and each person whose document contains the term adds the score to the calculator. After all terms are read, the persons with the highest scored documents present themselves. (This would be hard in the stadium analogy). (To match the real situation, we would actually need either 16 million people, or 256 documents and 256 calculators per person.)

The headlines for the documents with the highest total scores are then sent to the user. The user can view the text of each of these documents by clicking a mouse while pointing to the headline. When a user sees a document (or paragraph of a document) that answers his/her request, the user can mark the document "good" by pointing and clicking the mouse. The system collects all the terms from all the documents marked "good" along with all the initial words the user typed, and repeats the search process described above, but now using all these terms. Each search requires less than a second, even on databases up to 10 GBytes. This method, called "relevance feedback" [Salton 1972], generally produces a substantially better search than is possible with Boolean search systems [Blair & Maron 1986].

DowQuest DowQuest uses the method above, with enough memory so that a 1 GByte database is permanently stored. The 16 documents with the best scores are returned to the user, who can look at the full text (stored on a disk on the front end server) and can mark any document (or paragraph of a document) as "relevant". DowQuest has about six months worth of articles from nearly 300 different sources: Wall Street Journal, Washington Post, Barrons, the business sections of about 100 U.S. newspapers, Fortune, Forbes, and other magazines, and a number of trade publications (e.g. Byte). Ninety-nine percent of all searches take under 1.5 seconds, including front end time. CM time for 100 terms is about 170 msec. The system also parses input text, creates a new surrogate database, and updates the database while the system is operating on-line.

We have recently outlined algorithms that allow allow interactive access to 1 TeraByte (1,000,000 MBytes) of text [Stanfill, Thau & Waltz 1989]. For reference, the Library of Congress probably contains on the order of 40 Terabytes of text.

New Text Opportunities

This system suggests interesting opportunities for exploiting natural language processing results:

Adding NLP It is highly desirable to add natural language pre- and post-processing to the existing system, to improve its performance, and to extend its capabilities. For example, we are building recognizers that can find, label, and store lists of terms that refer to company names, geographic locations, names of persons, etc. Ultimately, this will help users to ask and obtain answers to questions that would be very difficult to phrase as Boolean queries. For example, "Earnings reports for New England utility companies" would expand to "Earnings reports for Maine, New Hampshire, Vermont, Massachusetts, Connecticut, Rhode Island, utilities, power companies, electric companies, power, light...". In addition, natural language processing systems will allow us to post-process retrieved documents, to filter out irrelevant articles, and thus improve the performance of the system from the user's point of view.

Adaptive Systems More intelligent processing could be applied to the users' queries; the system could keep track of user patterns and interests, and adapt itself to be easier to use or even to volunteer information it thinks the user is interested in.

WAIS (Wide Area Information Server) We are building a version of this system to seamlessly search an organization's local data as well as remote (e.g. Dow Jones) databasas. This system will allow organizations to locate and reuse proposals, reports and studies, find (by matching biographies) appropriate people for various tasks, and generally allow each person to locate any text - personal, corporate or public -- with a single search.

Hypertext The same methods that let us locate relevant articles can allow us to automatically build hypertext systems for text distributed on CD-ROMs. These could be generated much more quickly and inexpensively than is possible with hand-building.

Automatic Generation of NL Systems The retrieval system itself can be adapted to extract phrase, sentence and paragraph "templates" or patterns in order to aid the building of recognizers for particular topics or for types of stories. Such processing can provide empirical data on language usage that would be very difficult to find or invent any other way, leading to "dictionaries" of multi-word and multi-sentence language patterns and to FRUMP-like systems [DeJong 1982] with broad subject coverage.

Other related research, using dictionaries or thesauruses, has become popular in recent years. Some striking successes have been achieved by Ken Church and coworkers at AT&T Bell Laboratories [Church 1988, Ejerhed 1988] using the augmented "Brown Corpus" [Kucera & Francis 1982]. The Brown Corpus consists of one million words of text, chosen to represent a wide range of text types and styles (newspaper and magazine articles, books on history economics, etc.). It was "augmented" by Kucera and Francis by assigning each word in the corpus to one of about 450 classes, covering standard grammatical categories (noun, verb, adjective) but also including substantially finer distinctions (e.g.

noun-agent of sentence; verb-complements of particular types). Church collected statistics on the probabilities that various words would follow particular other word (or category) combinations. This system has been used to judge the most likely categories for words in novel text taken from news-wire sources. Success rates for Church's system are in the range of 98-99%, much higher than for the best syntactic parsers (in the range of 33% [Salton 1988]).

All these current lines of research emphasize *breadth* of coverage, rather than *depth* of coverage, and are thus complimentary to the goals of traditional AI-NL processing research. All present attractive alternatives to hand-coding [Lenat et al. 1986). And all can be used to accelerate the research into deep processing. The most attractive part of this effort is that our systems are immediately useful, and thus can pay for the research on their own augmentation. I believe these general approaches will have great importance in the ultimate story of the achieving of truly intelligent systems.

Memory-Based Reasoning

Methods broadly analogous to the text-search algorithms can be used to build "memory-based reasoning" systems to aid in decision-making. These systems perform like artificial neural nets [Rumelhart & McClelland 1986; Waltz & Feldman 1988] or ID3-like learning systems [Quinlan 1988]. In memory-based reasoning (MBR) a parallel machine is loaded with a database of the sort that can be used as a training set for learning systems: situations together with actions, classifications, or desired outputs for each situation. When a new problem is encountered, the MBR system compares it to all the known prior cases, and uses the most similar case (or majority vote of several similar cases) to classify the new case The key to successful MBR operation is the selection of a good similarity metric for matching new problems with known cases. [Stanfill & Waltz 1986].

Advantages of MBR MBR provides expert system-like behavior, but does not require extensive hand-coding. MBR provides "explanations" -- the precedents most similar to the current problem case -- in order to justify its actions. MBR provides high performance --superior to artificial neural nets [Wolpert 1989] and superior to expert systems (see below). MBR is robust when noise is added to its database; in one experiment [Stanfill & Waltz 1988] on the NETtalk database [Sejnowski & Rosenberg 1986], performance fell off only about 10% when 90% of the database was replaced with noise. MBR is simple to update: new cases can be added and old, obsolete ones removed, and performance will immediately track the changes. This is in sharp contrast to artificial neural nets, which must be totally retrained if the world changes, and expert systems, which are notoriously difficult to modify.

There are some disadvantages: MBR generally requires a data parallel computer, which will probably be more expensive than a system used to deliver an expert system or neural net application (though inexpensive data parallel systems without extensive interprocessor communication might suffice); and MBR systems do not operate as rapidly as a trained neural net, though they would generally be much faster than expert systems.

Classifying Census Data We have recently demonstrated that an MBR system can perform impressively on a task to generate one of about 241 industry codes and one of about 509 occupation codes for individual respondents, by comparing their answers (expressed as free text and multiple choice selections) with 132,000 cases that have already been classified by hand. Early results have indicated that by keeping categories where the system has been proven to be correct at least 90% of the time for industry codes and at least 86% of the time for occupation codes, MBR can correctly process at least 70% of the database for industry codes and about 56% of the database for occupation codes. For comparison, an expert system that required more than two years to develop, achieves only 57% and 37% of the database respectively on these two tasks [Smith, Masand, & Waltz 1990]; the MBR system took less than a month to build. For this application, the similarity metric is generated, using statistical operations, on the fly.

Other Applications Similar methods have been used to build MBR systems for optical character recognition, based on a large number of examples of handprinted numerals [Smith & Voorhees 1990]; for medical diagnosis [Stanfill & Waltz 1986]; for controlling a robot to produce near optimal trajectories [Atkeson 1987]; for (two-dimensional) object recognition [Tucker et al. 1988]; for automatically generating index terms for news articles or routing articles to appropriate recipients [Smith 1990]. Recent work [Zhang & Waltz 1990] on protein structure prediction has shown that a system that combines the results of MBR with neural nets and statistical information dramatically outperforms any previous method. This has relevance to the human genome project, another good target area for AI and IR. Many other applications are clearly possible.

Genetically-Inspired Methods

Market research is a "forest for the trees" problem. One needs to generate insights into the repetitive preference patterns among millions of customers, and distill market segment definitions in order to offer consumers the products they are most likely to want. We have developed genetically-inspired algorithms [Singer forthcoming, Holland 1975] that automatically find trends and categories without being told in advance what the patterns are.

One specific problem that has been addressed by these methods is the following: Suppose that we know purchasing behavior, demographic and credit information for several million (or tens of millions of) people, and that we wish to mail catalogs containing items selected from the offerings of hundreds of vendors, such that the greatest possible return (dollar amounts ordered minus the cost of the mailings) is maximized. Clearly, the larger the catalogs, the more the cost for postage, and the greater the chance each will be thrown away; the smaller and more tailored the catalogs, the better the return, but the more expensive it will be to print the catalogs and stuff the appropriate envelopes.

In an example run, we started with about 8000 customers on a small CM-2, and first calculated the ideal catalog of five items for each customer. (This step requires a model of consumer behavior.) Each list of five items is analogous to a piece of genetic material. We also computed an expected return (negative) for sending 8000 tailored catalogs. We then used the communications system of the CM-2 to randomly pair up consumers in parallel. For each pair, we then calculated the change in expected return if consumer 1 took consumer 2's catalog, and vice-versa. Both consumers were grouped into one or the other of the catalogs with a probability based on the change in the expected returns. The scoring scheme also makes it easier to merge a consumer who shares a catalog with a small number of others into a larger group than to pull a customer out of a large group and into a smaller one. Every few steps, random point mutations to the catalog's customers were probabilistically introduced, on the theory that the best catalogs may not have been present for <u>any</u> of the original consumers. The process continued until the maximum expected return point was found (in this case, 30 tailored catalogs, and sets of consumers who should receive each).

The overall solution to this problem required on the order of two hours on a 4K processor CM-2; the overall potential search space of solutions is on the order of 10^{87}!

A Different Route to Truly Intelligent Systems

So what does all this have to do with cognition? I want to argue that the basic associative memory operation of selecting relevant precedents in any situation is the essence of what intelligent entities do. ("Precedents" may be actions, options, remindings, etc.) If only a single precedent is found (e.g. when one is operating in a familiar environment on familiar tasks), then there is little involved in acting intelligently. Only when two or more incompatible precedents are found, or when the task space is unfamiliar, is reasoning (in the ordinary sense) required. Combinatorially explosive search can be avoided, since in any given situation only a small number of "operations" (actions) are plausible, making branching factors manageable. Planning can be supported by associative memory retrieval of precedents of the form: [hypothetical situation + goal + operator -> new situation] and/or [hypothetical situation -> goal]. Even creativity or analogical problem solving might be covered (if the best precedents match structurally but are not literal matches).

I am not imagining that a monolithic flat database could model memory. First, generalizations over memory and other structures need to be matched in addition to episodic items. (See [Kolodner 1989 and Evett et al. 1990] for descriptions of massively parallel frame systems.) Second, there ought to be situation-specific priming that changes the overall searchable space (or relevance judgements) for precedents. Overall, I am persuaded by society of mind [Minsky 1986] arguments and examples, and feel that the structure of memory also contains many agents responsible for recognizing special situations and either priming or censoring memories

Summary

If AI is to succeed, it is important to find ways to justify ongoing research costs, to substitute profits for promises. Commercial massively parallel applications already offer opportunities for changing the ways business is done, because existing limits on database size and speed of access can be transcended. Moreover, the excess processing capacity of massively parallel systems makes it possible to add greater intelligence to applications. And there are great potential payoffs for this kind of AI: even modest ideas, if they are sufficiently general to apply to an entire large database, can produce results that seem wonderfully magical. Most successes of this sort to date have used data parallel methods, especially memory-based reasoning. MBR applications can often be generated automatically from existing databases. MBR and case-based reasoning may also form the basis of new paradigms for cognition that can scale to human levels as massively parallel machines develop.

REFERENCES

Blair, D., & Maron, M. (1985). An evaluation of retrieval effectiveness for a full-text document retrieval system. *Communications of the ACM, 28*, 289-299.

Blelloch, G. E. (1986) "AFS-1: A programming language for massively concurrent computers." Tech. Report 918, Cambridge, MA: MIT AI Lab.

Church, K. (1988). A stochastic parts program and noun phrase parser for unrestricted text. Unpublished manuscript, AT&T Bell Labs, Murray Hill, NJ.

Dejong, G. (1982). An Overview of the FRUMP System. In W. Lehnert & M. Ringle (Eds.), *Strategies for natural language processing*, Hillsdale, NJ: Lawrence Erlbaum Associates.

Drumheller, M. (1986). "Connection Machine stereomatching". *Proceedings of the 5th National Conference on AI*, Philadelphia, 748-753.

Ejerhed, E. (1988). Finding clauses in unrestricted text by stochastic and finitary methods. Unpublished manuscript, AT&T Bell Labs, Murray Hill, NJ.

Evett, M., Hendler, J., & Spector, L. "PARKA: Parallel knowledge representation on the Connection Machine". Tech Report CS-TR-2409, University of Maryland, February 1990.

Fahlman, S. E. (1979). "NETL: A system for representing and using real-world knowledge". Cambridge, MA: MIT Press.

Kolodner, J. & Thau, R. (1988) "Design and implementation of a case memory", Research report. GIT-ICS-88/34, School of Information and Computer Science, Georgia Tech, October, 1988.

Kucera, H., & Frances, W. (1982). *Frequency analysis of English usage*, Boston: Houghton Mifflin Company.

Lenat, G., Prakash, M., & Shepherd, M. (1986). CYC: Using common sense knowledge to overcome brittleness and the knowledge acquisition bottleneck. *AI Magazine, 4*, 65-85.

Marr, D. C. & Poggio, T. (1976) "Cooperative computation of stereo disparity". *Science 194*, 283-287.

Quinlan, R. (1983) "Learning efficient classification procedures and their application to chess end games", in R.S. Michalski, J. Carbonell, and T. Mitchell (eds.), *Machine Learning: An Artificial Intelligence Approach*, Los Altos, CA: Tioga Publishing, 463-482.

Rumelhart, D., & McClelland, J., et al. (1986). *Parallel distributed processing*. Cambridge, MA: MIT Press.

Salton, G. (1971) *The Smart Retrieval System -- Experiment in Automatic Document Classification*. Englewood Cliffs, NJ: Prentice-Hall.

Salton, G. (1988). Personal Communication.

Schank, R.C. (1982). *Dynamic Memory*. Cambridge, UK: Cambridge University Press, 1982.

Schwartz, J.T. (1988) "The New Connectionism". *Daedalus 117, 1*, 123-142, Winter 1988.

Sejnowski, T.J. & Rosenberg, C.R. (1986) "NETtalk: A Parallel Network that Learns to Read Aloud". The John Hopkins University Electrical Engineering and Computer Science Technical Report JHU/EECS-86/01.

Smith, S. (1990) "Current limits of text-based intelligent systems", *AAAI Spring Symposium on Text-Based Intelligent Systems*, Stanford, March 1990, 130-133.

Smith, S., Masand, B., & Waltz, D. (1990) In preparation.

Smith, S., & Voorhees, H. L. (1990). In preparation.

Stanfill, C., (1988). Parallel computing for information retrieval: recent developments. Thinking Machines Corporation Technical Report #DR88-1.

Stanfill, C., & Kahle, B. (1986). Parallel free text search on the connection machine system. *Communications of the ACM, 29*, 12.

Stanfill, C., Thau, R., & Waltz, D. (1989). "A parallel indexed algorithm for information retrieval". *SIGIR '89 (Proc. 12th Annual International ACMSIGIR Conference on R&D in Information Retrieval)*, Cambridge, MA, June 1989, 88-97.

Stanfill, C. & Waltz, D. (1986) Toward memory-based reasoning. *Communications of the ACM 29*, 12, 1213-1228.

Stanfill, C., & Waltz, D. (1988). "The memory-based reasoning paradigm", *Proc: Case-Based Reasoning Workshop*, Clearwater Beach, Fl., May 1988, 414-424.

Tucker, L., Feynman, C., & Fritzsche, D. (1988). "Object recognition using the Connection Machine", *Proc. Computer Vision & Pattern Recognition*, Ann Arbor, MI, June 1988, 871-878.

Waltz, D. L. (1988). "The Prospects for Truly Intelligent Machines", *Daedalus 117*, 1, 191-212, Winter 1988.

Waltz, D., & Feldman, J. (Eds.) (1988). *Connectionist models and their implementations*. Norwood, NJ: Ablex.

Waltz, D. L. & Stanfill, C. (1988). "Artificial Intelligence Related Research on the Connection Machine". *Proc. International Conference on Fifth Generation Computer Systems*, Tokyo, December 1988, 1010-1024.

Waltz, D. L. (1989). "Is indexing used for retrieval?" *Proc.Case-Based Reasoning Workshop*, Pensacola Beach, FL., May 1989, 41-44.

Wolpert, D. (1989) "Generalization Theory, Surface-fitting, and Network Structures," Ph.D. Thesis, Physics Dept., University of California, Santa Barbara, Winter 1989.

Zhang, X., & Waltz, D. (1990) "Developing hierarchical representations for protein structures: an incremental approach". *AAAI Spring Symposium on Molecular Biology*, Stanford, March 1990, 149-152.