KNOWLEDGE-BASED SIGNAL INTERPRETATION

AT&T TECHNICAL JOURNAL

Sylvia D. Kuzmak, Mark D. Brittingham, Allen L. Gorin, Gregory A. Milich, and Joseph E. Shoenfelt

Sylvia D. Kuzmak, Mark D. Brittingham, Gregory A. Milich, *and* ***Joseph E. Shoenfelt*** *are in the Special Systems Design Department at AT&T Bell Laboratories in Whippany, New Jersey.* ***Allen L. Gorin*** *is in the Speech Research Department of AT&T Bell Laboratories in Murray Hill, New Jersey. Ms. Kuzmak is a member of technical staff. She is a principal investigator for a project applying knowledge-based system technology to signal interpretation problems. She received an A.B. in statistics from Princeton University, an M.A. in mathematics and education from the University of Chicago, and a Ph.D. in psychology from the University of Pennsylvania. She joined AT&T in 1984. Mr. Brittingham is a member of technical staff, responsible for development of inference systems. He received*
(continued on page 120)

Signal interpretation is the process of using signal data to develop a high-level description of an environment, including the objects present, their classifications, and their locations. Signal interpretation plays a central role in surveillance systems whose task is to detect, classify, and localize specific platforms (such as airplanes and ships) on the basis of signals they emit or reflect. AT&T has been working in the area of surveillance technology for 40 years and is continuing work to improve detection performance and to meet demands for processing data from distributed sensor systems. This paper describes how artificial intelligence (AI) technology is being integrated with knowledge and algorithms from previous AT&T programs to develop new signal interpretation systems for distributed sensor systems. These new systems extend platform identification and classification capabilities, fuse information over space and time, enhance system modifiability, and extend the capabilities of the human-machine interface. Plans for exploiting the ASPEN parallel processor to meet real-time processing demands are also described.

Signal Interpretation

Consider the problem of detecting and classifying objects using a distributed sensor system. Such a system comprises many individual sensors that are geographically distributed. Objects to be detected emit or reflect signals that supply information about the location and type of the object in question. The receptions of these emissions from the individual sensors are then all transmitted to a central processing facility for further machine processing and analysis by human operators. The

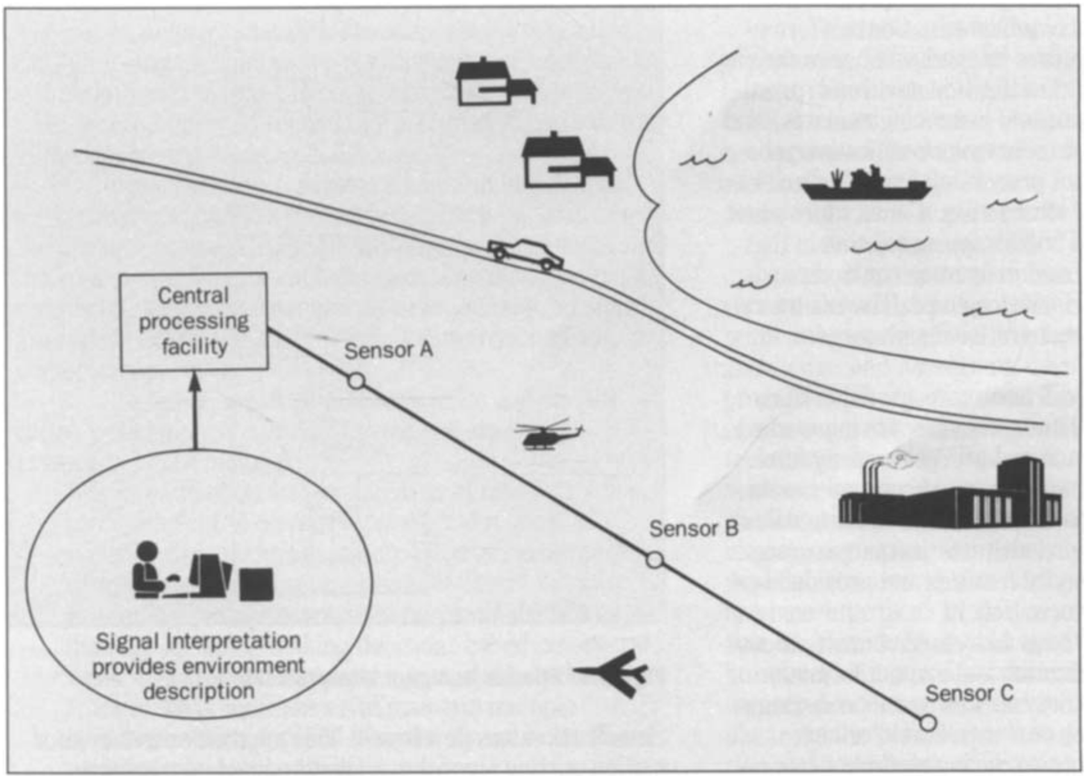

Central processing facility
Sensor A
Sensor B
Sensor C
Signal Interpretation provides environment description

Figure 1. A surveillance system with distributed sensors.

machine processing involves signal interpretation, that is, developing a high-level description of the environment within the sensor field based on the signal data. This description includes the objects detected, their classifications, and their locations, and can be presented using a geographic plot. Human operators review the description of the environment for potential objects of interest. Operators then access explanations and review the data on which conclusions are based to reach their own conclusions regarding those objects. Automatic processing serves to direct and expedite operators' surveillance activity. Figure 1 illustrates a general surveillance system.

Signal interpretation can be divided into three phases: signal processing, signal characterization, and contact processing (see Figure 2). During signal processing, raw sensor data is digitized, enhanced, and smoothed. During signal characterization, thresholding is applied to identify signal detections and remove interference and noise, continuity of detections over time is established, and basic features of detections (such as signal-to-noise ratio and physical location) are estimated.

Contact processing completes the job of developing the high-level environment description. Two main functions of contact processing are contact formation and classification. During contact formation, detections are clustered into groups believed to be originating from the same platform. The cluster of detections is called a *contact*. During classification, each contact is assigned a set of possible classifications with confidences. Contact formation and classification processes interact in a bootstrapping fashion in that preliminary classification decisions regarding individual detections are used in forming contacts, and then contacts are subjected to further classification processing. An additional contact processing function discussed only briefly in this paper is localization. Contact formation and localization also have a bootstrapping relation in that localization information is used in forming contacts, and then contacts, once formed, are localized. The main focus of this paper is on the contact processing phase of processing.

AT&T Experience. AT&T has extensive experience within each phase of signal interpretation, having worked in some areas for 40 years. Signal processing programs have developed algorithms and advanced computer architectures that provide a capability to detect weak signals in noise and interference. Signal characterization programs have identified important signal features and provided systems for automatic characterization.

Several programs have addressed contact processing issues, including classification and contact formation. An early approach to classification was based on discriminant analysis. The approach performed well for high signal-to-noise ratio contacts producing multiple detections, which led to the decision to pursue further the development of automatic contact processing. The use of a discriminant analysis approach, however, increased awareness of the importance that systems be easily updatable and capable of generating explanations, both of which were difficult using that approach.

Early experience with automatic contact formation revealed the difficulty of making contact formation decisions, especially for low signal-to-noise ratio contacts. The early experience showed the importance of maintaining multiple hypotheses regarding fusion of detections until conclusive evidence is obtained, since false fusion adversely affected classification and localization.

To handle low signal-to-noise ratio contacts not formed into multidetection contacts, another approach to classification was developed. This approach used an information-sorting algorithm applied to single detections, which enhanced performance for signal-detection contacts. This algorithm provides the foundation for classification of individual detections in the current systems.

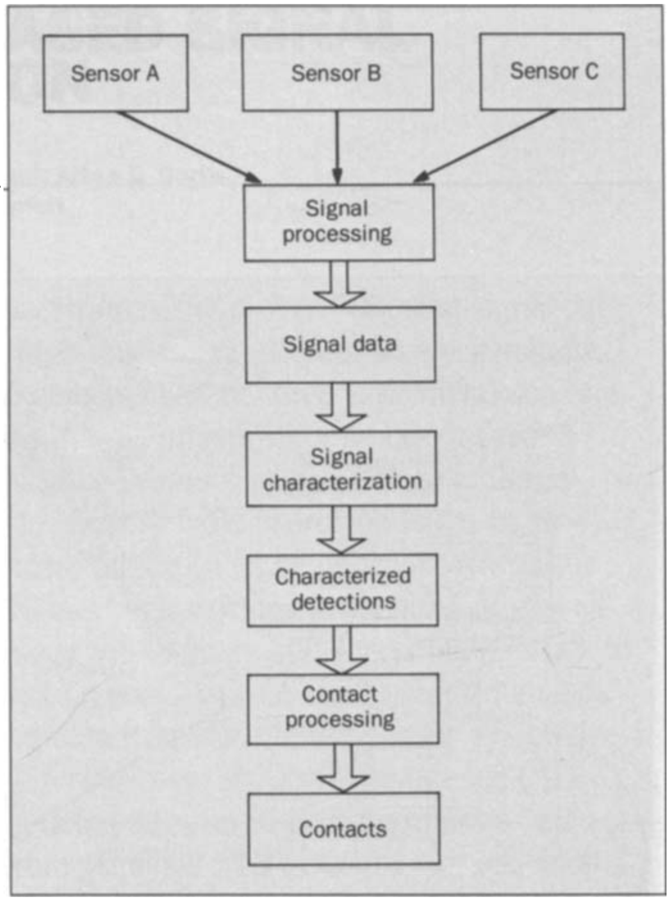


Figure 2. Phases in signal interpretation.

Contact processing approaches have been tested extensively with real data, often in field settings, which has led to the accumulation of valuable knowledge regarding the effectiveness of different approaches. Field experience has led to an understanding of the importance of allowing system modifications, e.g., to permit tailoring for particular geographic sites, to address changes in the expected distribution of platform classes, or to include the capability to handle a new platform class.

AT&T's extensive experience with surveillance systems played a significant role in the decision to incorporate AI technology into current systems to provide such features as enhanced software modifiability, explanation capability, and techniques for reasoning under uncertainty and maintaining multiple hypotheses. AT&T's current systems incorporating AI technology are processing real data in the laboratory.

Current Challenges. There are continuing demands for improving signal interpretation technology, including the quality of the environment description that results and the speed with which it is generated. The development of distributed sensor systems poses new demands on signal interpretation systems to coordinate information from numerous sensors.

To improve platform identification and classification performance, several approaches are being explored. These include:

1. The identification and exploitation of new signal features relevant to platform identification and classification, including features that are emergent from spatial and temporal fusion of data
2. Extending the use of multiple factors in decision-making, including making decisions based on several weak indicators (this assumes a capability to keep track of weak indicators and maintain multiple hypotheses with degrees of certainty until conclusive evidence can be identified)
3. Developing a more complete environment description, including uninteresting platforms as well as interesting platforms, so that the interpretation of an individually ambiguous detection can be clarified by evaluating its relations to both interesting and uninteresting platforms.

With the increase in the variety of factors used to make contact identification and classification decisions, the importance of providing an explanation capability for a human operator is increased.

The development of distributed sensor systems leads to a need to coordinate information from various sensors, both to accumulate evidence to use in decision-making and to organize information for presentation to a human operator to reduce operator workload.

The potential for rapid changes in the population of platforms, including the addition of new platform classes, which would then necessitate changes in how features are used in contact identification and classification, has increased the importance of software modifiability. In addition, adapting the system for different sensor types or deployment sites requires software modifiability.

Current Systems. AT&T Bell Laboratories is developing new signal interpretation systems to address the current demands on surveillance systems. The new systems integrate artificial intelligence technology with knowledge and algorithms developed through previous programs. This paper describes how, in systems currently under development, artificial intelligence technology is being used in conjunction with previously acquired knowledge to enhance contact identification and classification performance, to support the fusion of information over space and time, to enhance system modifiability, and to extend the capabilities of the human-machine interface. In this paper, examples of the application of AI techniques and the role of previous knowledge are provided for contact formation, classification, and contact processing software architecture. These examples illustrate the hybrid approach to knowledge representation and knowledge utilization procedures used in current systems to incorporate different problem-solving approaches appropriate for different aspects of system processing. In addition, work on incorporating the ASPEN parallel processor to achieve real-time performance is described.

Contact Formation

Contact formation refers to the process of grouping detections believed to be originating from the same platform. Previous systems have addressed the problem of grouping detections at a sensor over time as coming from the same platform (contact formation at a sensor). Current systems extend the process to include spatial and temporal fusion across a distributed field of sensors (contact formation across the field). They also provide for software modifiability, generation of explanations, reasoning under uncertainty, and maintenance of multiple hypotheses.

The wider scope of contact formation can enhance system detection and classification performance by (1) permitting the exploitation of new features relevant to classification that emerge from the spatial and temporal fusion of data, and (2) allowing the accumulation of evidence for classification over space and time. As an example of the former, if two sets of detections on different sensors at different times are grouped as coming from the same platform, sensor spacing and time difference can be used to estimate the "emergent" platform speed of advance. This feature, in turn, can be used to refine the classification. As an example of the effect of the accumulation of evidence, if detections on various sensors are grouped as coming from the same platform, any revealing feature among the group may be used to classify the group. In this way, ambiguous detections may be classified through association with unambiguous detections. A case of this would be to identify an individually ambiguous detection as being from an uninteresting platform, through association with detections that were clearly from an uninteresting platform. Recall that this sort of phenomenon motivates keeping track of both interesting and uninteresting platforms. As another example of the effect of the accumulation of evidence, weak indicators of classification among a group of detections may combine to provide a conclusive classification for the group.

The wider scope of contact formation to include contact formation across the field also has the potential to facilitate human-machine interaction by organizing and reducing information presented to the operator. Data from a distributed sensor system with many sensors can quickly overload an operator. Organizing detections into contacts limits the number of separate items presented to the operator, and thus reduces workload. In addition, information fusion across the field of sensors supports contact localization, e.g., by permitting the combination of concurrent indications of location from different sensors to obtain a better localization.

Contact Representation. The potential of contact formation processing to improve overall system performance has been described. To support contact formation, a means for representing a contact, which is the result of signature, temporal, and spatial fusion of detections, is required. ("Signature" formation at a sensor refers to clustering detections believed to be from the same platform.) Detections cannot merely be clustered into an unstructured group, but must be organized in a structured representation that records the relationships among detections, including signature, temporal, and spatial relations. Information on these relationships must be preserved to permit identification of emergent features or to support the accumulation of evidence over space and time.

The representation facility provided by symbolic languages such as Lisp provides a natural means of realizing these representations. A basic characteristic of this representational capability that is particularly appropriate for contact representation is the ability to define symbols, relations between symbols, and information qualifying the relations between symbols,[1,2] including confidence in the relation and supporting evidence. In our current system, symbolic contact representation is accomplished using Lisp and Flavors, an object-oriented programming extension to Lisp.[3] Object-oriented programming is described under "Software Architecture."

The representation of a contact using a set of objects and relations is illustrated in Figure 3 (the representation shown has been simplified for expository purposes). Object representations are defined for individual detections, sensor contacts (groups of detections at a sensor), and field contacts (groups of sensor contacts across the whole sensor field). There are signature component relations between detections and sensor contacts; sensor contact component relations between sensor contacts and field contacts; and temporal or regain relations from detection to detection, sensor contact to sensor con-

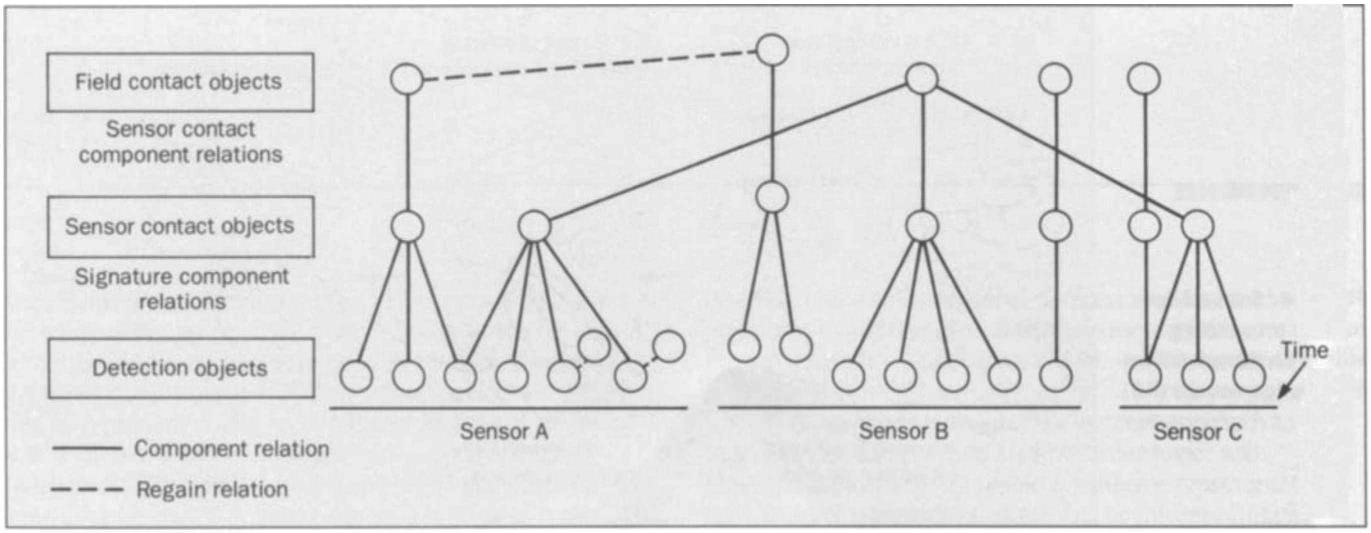


Figure 3. Contact representation objects and relations.

tact, and field contact to field contact. A regain relation occurs between an object that has been lost and another object that reappears later in time and seems to be the same source.

Although not illustrated in Figure 3, each relation between objects has a confidence value associated with it, recording the confidence that the two objects are related in the particular fashion. In this way, uncertainty regarding contact formation decisions is represented when there is only weak indication of relation. Multiple conflicting hypotheses can be represented using multiple low confidence relations, as would be the case for the detections in the figure that have signature component relations to two distinct sensor contacts.

In addition to a confidence value, for each relation in the contact representation there is also information recorded regarding supporting evidence for the relation. This information can be used to generate explanations for contact formation decisions.

Contact Formation Processing. Providing a complete description of the process of building contact hypotheses is beyond the scope of this paper. However, in general, the contact formation process can be characterized as both hierarchically incremental and largely homogeneous among increments, as described below. Within the contact formation process, knowledge and techniques from previous systems are integrated, as described below.

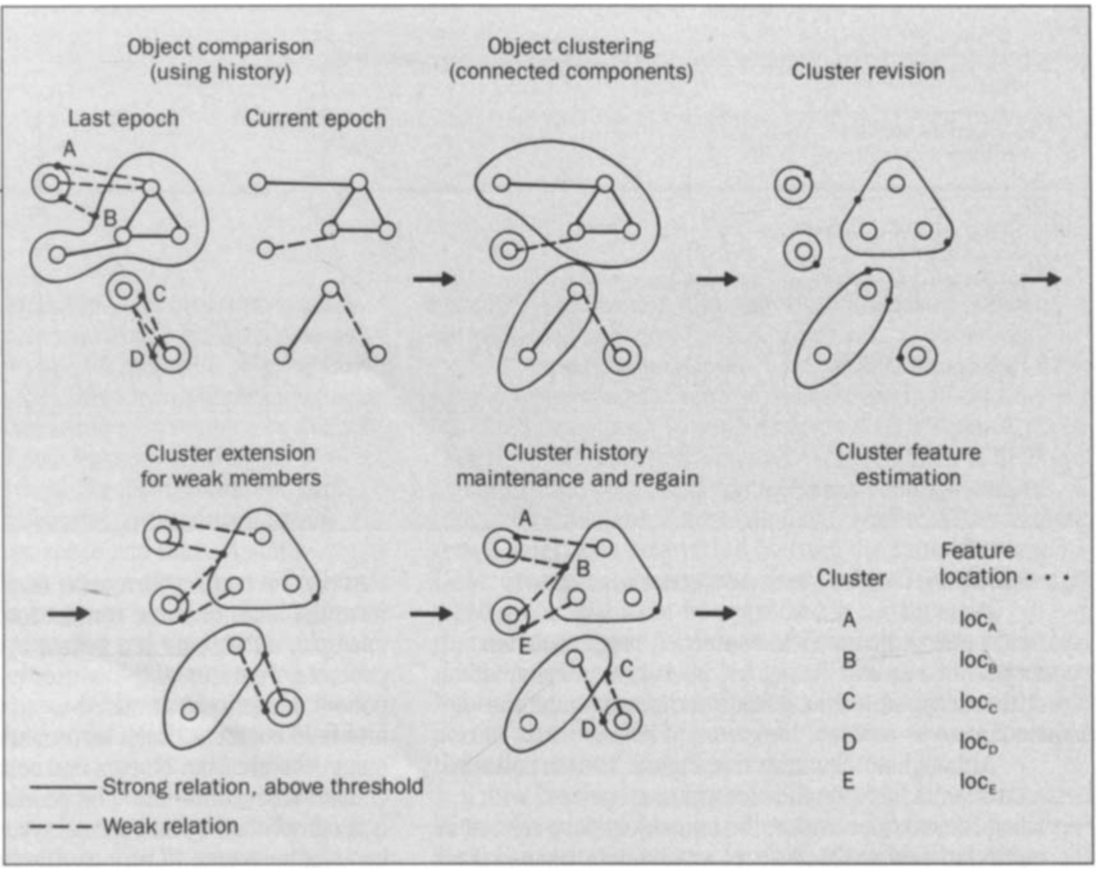
The contact formation process is incremental in that certain contact formation decisions are made first, and form the basis for later contact formation decisions. For example, detections at a sensor are grouped into sensor contacts. Features of the sensor contacts (and their component detections) are used to group the sensor contacts into field contacts. Each increment corresponds to developing a level of the objects and relations in the hierarchical contact representation. The processing in each increment is accomplished by a separate system functional unit or knowledge source (KS): e.g., the sensor contact formation KS or the field contact formation KS.

Contact formation is largely homogeneous among increments (or KSs) in that each increment involves similar processing, including phases of:

1. Object comparison
2. Object clustering
3. Cluster revision
4. Cluster extension for weak members
5. Cluster history maintenance and regain
6. Cluster feature estimation.

The homogeneous nature allows one uniform approach to explaining the decisions for the various increments of contact formation, which provides for simplification of the human-machine interface. It also encourages software compactness, modifiability, and maintainability. Each of the steps in contact formation within an increment is illustrated in Figure 4, and described in more detail below.

Object comparison involves the pairwise comparison of objects to be clustered for indications that each pair of objects is related; e.g., for each pair of sensor contacts



Object comparison (using history)
Last epoch
Current epoch
A
B
C
D
Object clustering (connected components)
Cluster revision
Cluster extension for weak members
Cluster history maintenance and regain
A
B
E
C
D
Cluster feature estimation
Cluster
Feature location • • •
A locA
B locB
C locC
D locD
E locE
Strong relation, above threshold
Weak relation

Figure 4. Contact formation processing within an increment or knowledge source (KS).

in the field, look for indications that the sensor contacts are from the same platform. (Heuristics can be defined to restrict the comparisons between pairs of sensor contacts, e.g., on the basis of sensor proximity. However, for simplicity, this case is not considered.) Past experience with classical clustering algorithms in signal interpretation systems has led to the identification of many factors relevant to such decisions. These factors are incorporated in a set of tests that are applied to each object pair. Tests may refer to any features of the two objects, including their past history of relation to each other. When a test is passed, a weight of confirming evidence for the relation between the objects is recorded. Weights for each test that passes are combined to obtain a total confidence in the relation between the two objects. Object comparison results in the identification of a set of relations and confidences among the objects. If a threshold confidence is assumed, the result can be represented as a graph with the nodes representing objects and the edges representing relations above a threshold confidence. Without thresholding, the result can be represented as a labeled graph.

Object clustering involves the application of a mathematical clustering technique to the graph identified during object comparison (e.g., to identify sensor contacts in the field that represent the same platform). Several clustering techniques have been used in previous systems. The exact clustering algorithm incorporated in the current systems is modifiable. However, the current algorithm is "connected components," by which an object is placed with a cluster if it is related above threshold to any object in the cluster.

Cluster revision modifies these initial clusters (e.g., candidate field contacts) using heuristics, based primarily on characteristics of the clusters as a whole. Heuristics for cluster revision at the sensor contact level have been identified in previous programs and are incorporated in the current system.

Cluster extension to include weak members involves, for each cluster (e.g., field contact), identifying objects (e.g., sensor contacts) for which there is only weak evidence (through the object comparison tests) for belonging with the original "core" objects of the cluster. A numerical "degree of belonging" of a weak member to a cluster is calculated on the basis of such evidence. Being able to represent confidence in the belonging of an object with a cluster provides a capability to represent uncertainty in contact formation decisions. For any given object, there may be multiple clusters to which there is some possibility it belongs.

Cluster history maintenance and regain compares the current cluster (e.g., field contact) with clusters (e.g., field contacts) from the previous time epoch, and decides which clusters are continuations of others. To the extent that object history was used in determining relations between objects, current clusters will tend to be biased to reproduce previous clusters. Decisions regarding which component objects are regains or continuations of lost component objects for the cluster are also made.

Finally, feature estimates (e.g., location) for the revised cluster are updated on the basis of the current epoch's information.

Aspects of contact formation processing that can be easily modified because of modular design include, but are not limited to, specific tests during object comparison, the clustering algorithm applied during object clustering, and heuristics used in cluster revision.

In cases of large numbers of sensors and many detections per sensor, contact formation processing as described here is very time-consuming. For this reason, work investigating the potential for applying parallel processing to contact formation is being pursued. This work is described in more detail under "Parallel Processing."

Classification

The classification subsystem integrates AI technology and techniques from previous programs to make classification decisions. Contacts are classified using a logic-based inference procedure that generates multiple hypotheses (with confidences) selected from a hierarchy of possible classification hypotheses. This procedure is described below. Individual detections are classified as a precursor to contact classification using a modification of the information sorting algorithm referred to under "AT&T Experience."

Logic-Based Inference. The current approach to contact classification is based on first-order logic[4] with extensions for quantitative and qualitative representations for the uncertainty in the domain. The inference engine developed for reasoning in the domain features a facility for integrating user interaction into the reasoning stream at any point, a flexible control structure, the ability to generate and retract assumptions, and a limited capacity for meta-level reasoning (illustrated below). In addition, the processing of rule conditions incorporates conjunctions, disjunctions, preferential matching, exceptions, negations, hooks into the Lisp system environment, or any nested combination of the above.

In first-order logic, the primary mechanical technique for generating conclusions from a given set of facts and inference rules is known as resolution theorem proving.[4] There is a technique derivable from "set of support" resolution theorem proving, known as deductive retrieval, that plays an important role in many logic-based inference engines. In deductive retrieval, an inference engine begins with a form to be proven and traces a chain of rules back to a set of facts in the database. If a chain such as this can be built on the rules found in the system, or if the request is supported by the database directly, then the goal form is proven.

For example, in Figure 5 the database holds facts relevant to a single contact (and its component detections). The rules below the database are designed to prove that a given contact is a jet. Starting with the hypothesis that the contact is a jet, the system works backward through rules 1 and 3 and concludes that this hypothesis is supported by the facts in the database. Note that the system knows to call a given rule because the conditions of the goal rule can be matched to the conclusions of the called rule. The rule used in this example states that one way of showing that a contact is a jet is to find one detection corresponding to an

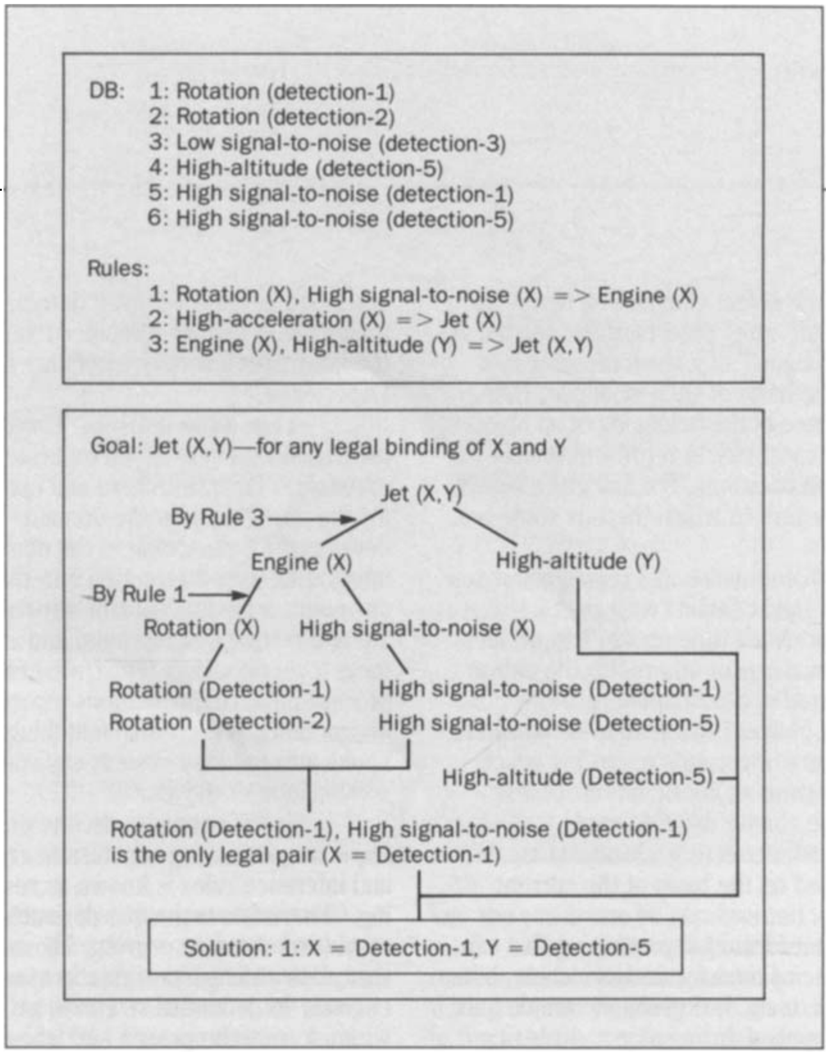

DB: 1: Rotation (detection-1)
2: Rotation (detection-2)
3: Low signal-to-noise (detection-3)
4: High-altitude (detection-5)
5: High signal-to-noise (detection-1)
6: High signal-to-noise (detection-5)
Rules:
1: Rotation (X), High signal-to-noise (X) => Engine (X)
2: High-acceleration (X) => Jet (X)
3: Engine (X), High-altitude (Y) => Jet (X,Y)
Goal: Jet (X,Y)—for any legal binding of X and Y
Jet (X,Y)
By Rule 3
Engine (X)
High-altitude (Y)
By Rule 1
Rotation (X)
High signal-to-noise (X)
Rotation (Detection-1)
Rotation (Detection-2)
High signal-to-noise (Detection-1)
High signal-to-noise (Detection-5)
High-altitude (Detection-5)
Rotation (Detection-1), High signal-to-noise (Detection-1) is the only legal pair (X = Detection-1)
Solution: 1: X = Detection-1, Y = Detection-5

Figure 5. Chaining backward from a goal through rule conclusions to consistent entries in a database.

engine and another detection indicating high altitude. For a detection to correspond to an engine, it must have evidence of rotation and high signal-to-noise ratio. Since detection-1 fits both of these descriptions, it is bound to the variable *X*. No other detection fits this description, so no other detection gets consistently bound to this variable. Since detection-5 has the predicate "high-altitude," it is bound to the variable *Y*. These are the only two legal and consistent bindings that prove the original goal.

Dealing with Uncertainty. In the domain of signal interpretation, statements, rules, and observations are frequently characterized by uncertainty. A scheme is required to cope with this uncertainty. A common way to deal with uncertainty in logic-based systems is to assign a confidence *p* to each hypothesis and rule in the knowledge base. When inference procedures draw conclusions from the information in the knowledge base, they combine the confidences of each of the supporting hypotheses as well as the implicational confidence of the rule that brings them together. Thus in Figure 5, the database hypothesis "rotation (detection-1)" may be stored with a confidence of 0.6. This implies that the evidence supports the belief that detection-1 is produced by a rotating source only to the extent summarized in the numerical term 0.6. The rule associating high signal-to-noise ratio and rotation (rule 1) may be only 0.5 confident. The uncertainty combination system needs to handle both the confidence in the original hypothesis and the confidence in the rule's association to

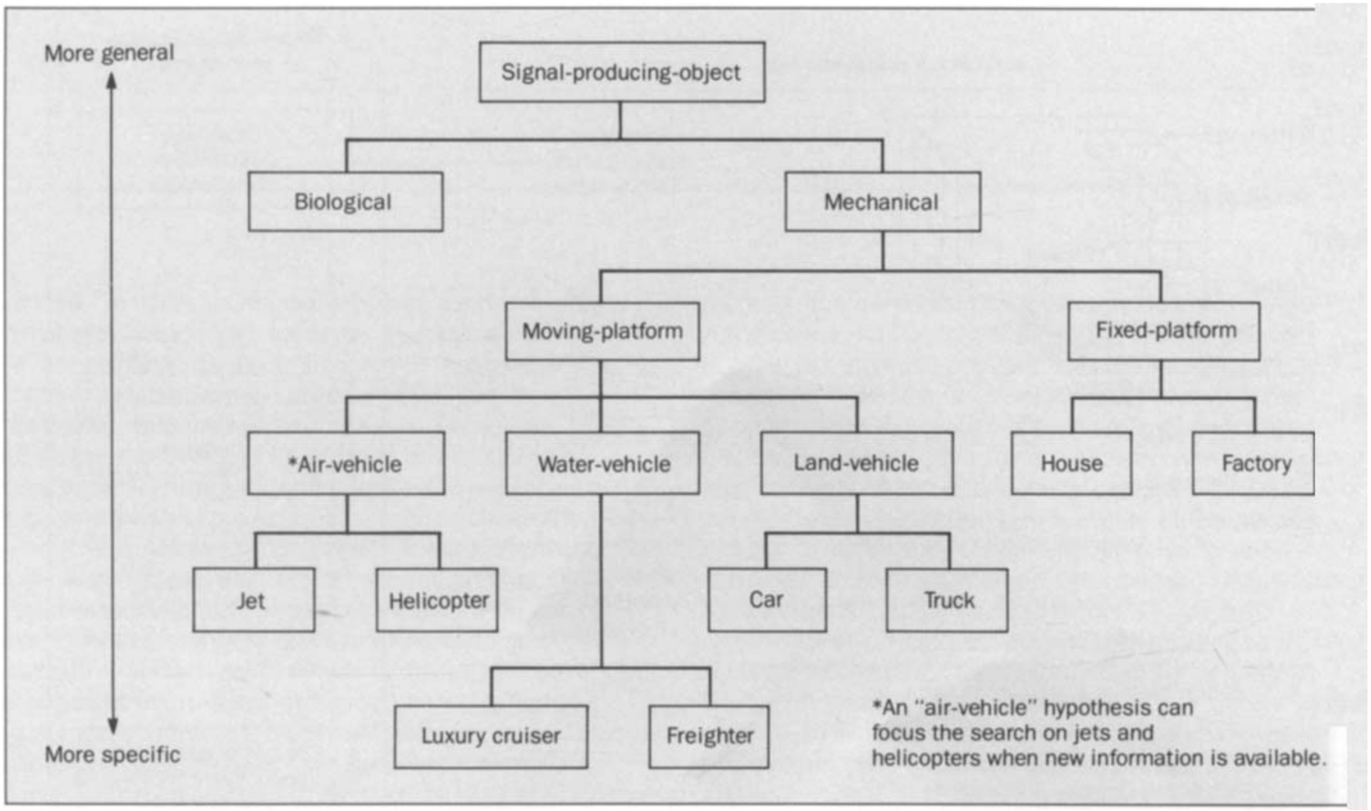


Figure 6. A classification hierarchy with search-limiting heuristics.

derive the confidence in the resulting hypothesis.

An alternative to the use of numerical measures of uncertainty is found in the theory of endorsements.[5] Instead of summarizing uncertainty in a single number, an endorsement provides a qualification for a hypothesis or rule. That is, it specifies the reasons why something should be believed or disbelieved. Thus information is not lost when combining hypotheses and rules; rather it is passed along for further use by the inference engine. An example would be a "high-altitude" endorsement for a "jet" hypothesis. If it later becomes apparent that the source was not at a high altitude, it may be possible to transfer the confidence in the classification directly to other possible classifications.

The present approach to uncertainty makes use of a combination of quantitative and qualitative methods. For any decision point in which information about uncertainty is to be propagated, a numerical confidence is calculated in an appropriate manner. After this procedure, however, a second operation occurs in which symbolic uncertainty values are inspected and manipulated. At this point a range of options are available. The numerical confidence may be changed, new qualitative values may be added or old ones deleted, an arbitrary procedure may be called to query the user for information or otherwise attempt to reduce the uncertainty, or control procedures may be invoked.

The Class Hierarchy. The range of platform classifications that can be inferred from the input data are represented in the system by a class hierarchy. General classes, such as biological objects or mechanical objects, are found near the root of the hierarchy while specific classes, such as specific types of air, water, and land vehicles, are found at the leaves (see Figure 6). The class hierarchy is used to organize classification rules and their application and to facilitate the propagation of confidence among classification hypotheses for a contact. In addition, a model of the object represented by a node in the hierarchy is maintained on the node. The system operates by utilizing preliminary classifications (from individual detection classifications) of the contact object to guide the choice of a set of possible classes in the class hierarchy. Those

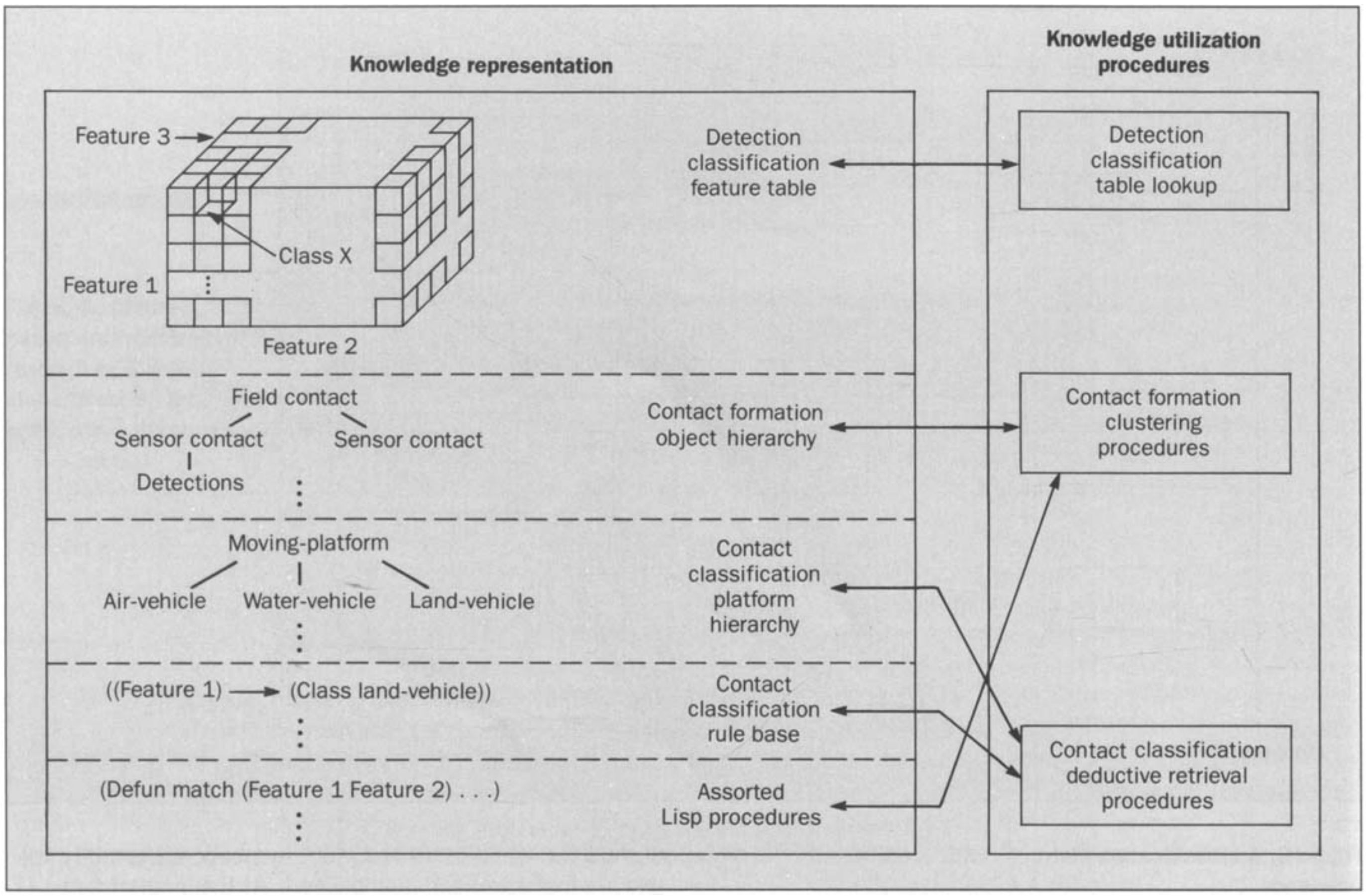

Knowledge representation
Knowledge utilization procedures
Feature 3
Class X
Feature 1
Feature 2
Detection classification feature table
Detection classification table lookup
Field contact
Sensor contact
Sensor contact
Detections
Contact formation object hierarchy
Contact formation clustering procedures
Moving-platform
Air-vehicle
Water-vehicle
Land-vehicle
Contact classification platform hierarchy
((Feature 1) → (Class land-vehicle))
Contact classification rule base
Contact classification deductive retrieval procedures
(Defun match (Feature 1 Feature 2) . . .)
Assorted Lisp procedures

Figure 7. Hybrid knowledge representation and utilization procedures.

classes that are members of this set are then submitted to the inference engine as hypotheses to be proven along with the detections from the contact and the rules from the class node. When a specific class has received support from this process, this support is then propagated up the class hierarchy to the more general classes. Furthermore, when new information about an object arrives, it is possible to focus the attention of the system to the more specific hypothesis levels below the best previous hypothesis (see Figure 6). This focusing capacity, along with an ability to add or delete rules from a rule set in response to the tactical context and current data, represents the system's meta-level reasoning.

Classification rules incorporate parametric information (as in previous systems) as well as additional heuristics. In many cases, purely mathematical descriptions have been replaced by representations that are more natural and intuitive for the knowledge engineer and domain expert, which facilitates the rule generation process.

The overall result of the classification effort is a flexible and powerful system for implementing inference about signal-based classification. The rule-based aspect of the system permits the efficient alteration of knowledge about contact classes and the generation of explanations of reasoning to aid the human user.

Software Architecture

The contact processing software must incorporate many diverse types of knowledge to support the various system functions, including contact formation, contact classification, and localization. Attempting to represent and utilize such diverse knowledge using only one knowledge representation framework, such as production rules, is difficult. In this system, knowledge representation is not limited to one framework, but rather, a hybrid approach is adopted,[6] in which knowledge representation frameworks that appear natural for a particular problem are selected for that problem. As shown in Figure 7, knowledge is represented in many forms, including tables, object hierarchies, rules and Lisp procedures. In addition, there are different procedures for utilizing this knowledge, including a clustering algorithm for contact formation and a deductive retrieval procedure for contact classification.

The software architecture for our current contact processing systems uses object-oriented programming.[3] Object-oriented programming involves the definition of structures called objects that have associated properties (attributes) and actions (methods or messages). Generally, all pertinent information, including attributes, methods to update and retrieve attributes, and other methods important in describing an object are stored directly with that object. Control is realized by message passing among the objects. This programming technique has several advantages, including facilitating the development and maintenance of large software systems.[7] Object-oriented programming also provides a foundation for building object hierarchies, such as the contact formation object hierarchy and the contact classification platform hierarchy.

The object-oriented software architecture for our current contact processing systems is illustrated in Figure 8. It has been implemented on a Symbolics Lisp machine using Common Lisp and Flavors. Object representations are used for domain objects, including detections, sensor contacts, field contacts, sensors, and field, as well as for the controller and knowledge sources (KSs), including the Detection Classification KS, Sensor Contact Formation KS, Field Contact Formation KS, Contact Classification KS, and Localization KS.

As illustrated in Figure 8, overall contact processing is controlled by the controller object that sends specific task requests to domain objects. A task request is a message sent by the controller object to a domain object, requesting it to perform a task. The controller's strategy attribute contains the global problem-solving strategy that specifies the order in which domain objects receive task requests. Once a domain object receives a task request it is responsible for locally deciding what specific knowledge to apply to handle the request. It then sends a knowledge request message to a KS, requesting it to execute. For example, when the controller sends a contact formation request to a sensor, the sensor object receives the message and sends a knowledge request message to the Sensor Contact Formation KS. A feature of this message request hierarchy is that it provides for a KS to be selected depending on state information. For example, several Sensor Contact Formation KSs can be defined that are appropriate under different contexts. Depending on the sensor type and other state information, the appropriate KS can be selected by the sensor.

To facilitate real-time processing on the Symbolics Lisp machine, the system architecture is multitasking (see Figure 9). Concurrent process execution is especially important during real-time processing of field data. In addition, virtual memory must be periodically reclaimed without drastically affecting the performance of the system. To accomplish this, an object purge and maintenance procedure has been integrated with the Symbolics ephemeral garbage collection process.

Parallel Processing

Contact Formation and Parallel Processing. Contact formation has been identified as an area of contact processing that requires parallel processing. One reason is that it is a computationally complex process requiring orders-of-magnitude acceleration over a uniprocessor to achieve real-time execution for cases of sensor fields with many sensors and heavy traffic (many detections per sensor). In addition, there is much algorithm development left to be done for contact formation, requiring a programmable computer rather than a fixed hardware implementation. Accelerating the execution of contact formation facilitates algorithm development by broadening the volume of data on which the algorithm can be evaluated.

Computational Complexity. The two most computationally intensive stages in contact formation are object

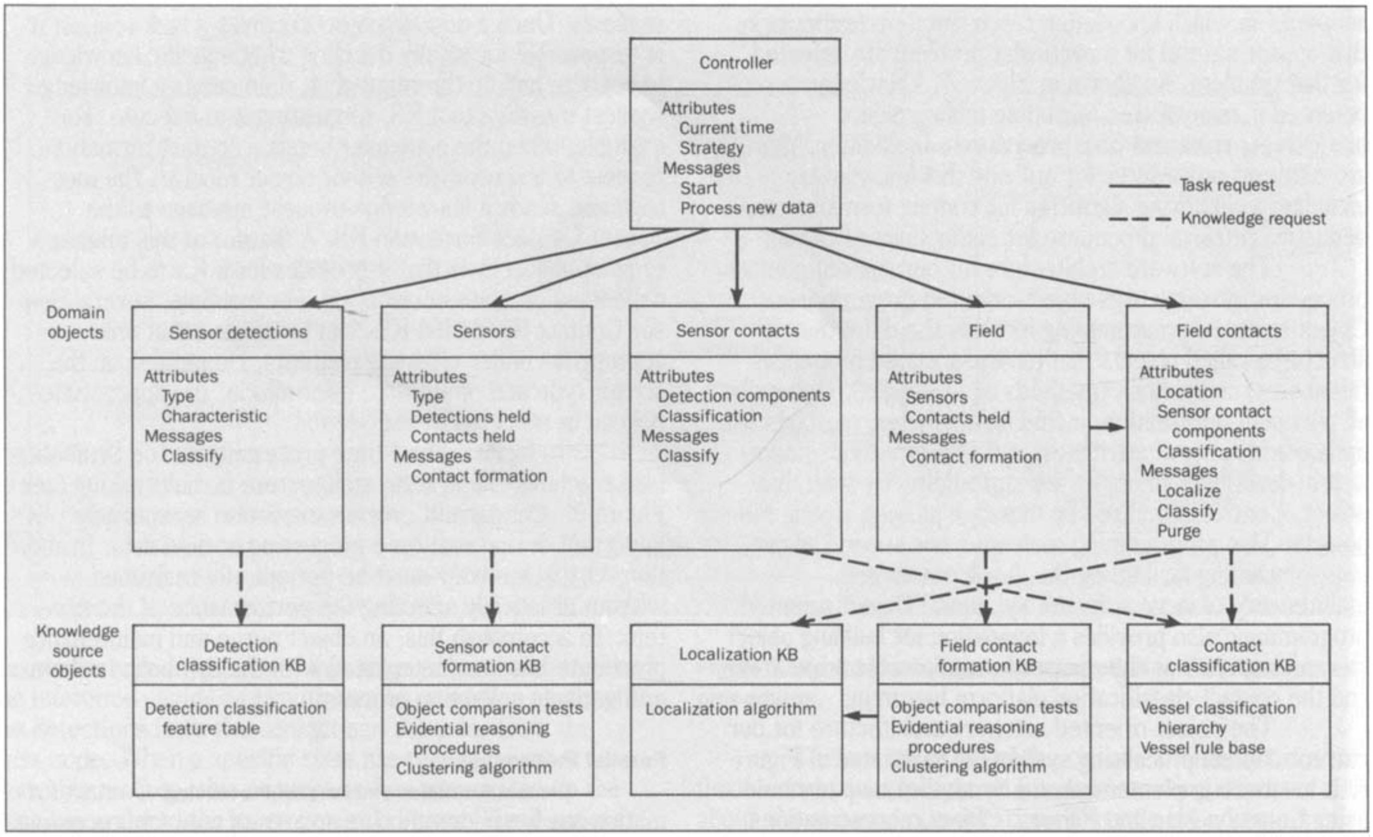


Figure 8. Object-oriented contact processing software architecture. KB = knowledge base.

comparison and object clustering. The first is an all-pairs comparison that involves comparing all objects with one another (e.g., sensor contact to sensor contact throughout the field). This process of computing the relationship graph in general increases quadratically with the number of objects. The second stage is to partition the objects into clusters (e.g., sensor contacts into field contacts) on the basis of those relationships. There are a variety of mathematical clustering procedures that can accomplish this partitioning process. The complexity of the partitioning process depends on the specific procedure being utilized. In particular, the connected-component partitioning algorithm has complexity that is linear in the number of relationships. Since the number of relationships is bounded by the square of the number of objects, this implies that the two stages together have quadratic complexity.

Applying this analysis to field contact formation, we find the complexity of the process grows quadratically with the number of sensor contacts. Since the number of sensor contacts is related linearly to the number of sensors, the field contact formation process has complexity

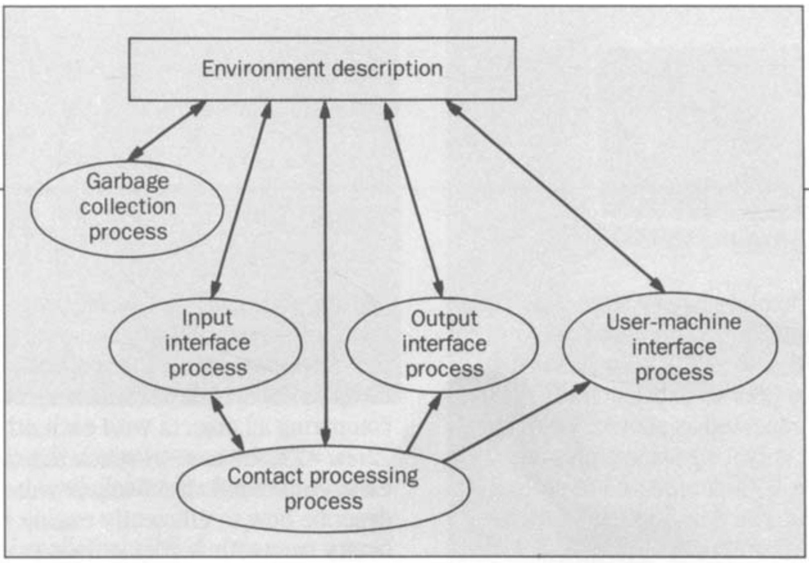

Environment description
Garbage collection process
Input interface process
Output interface process
User-machine interface process
Contact processing process

Figure 9. Multitasking facilitates real-time performance.

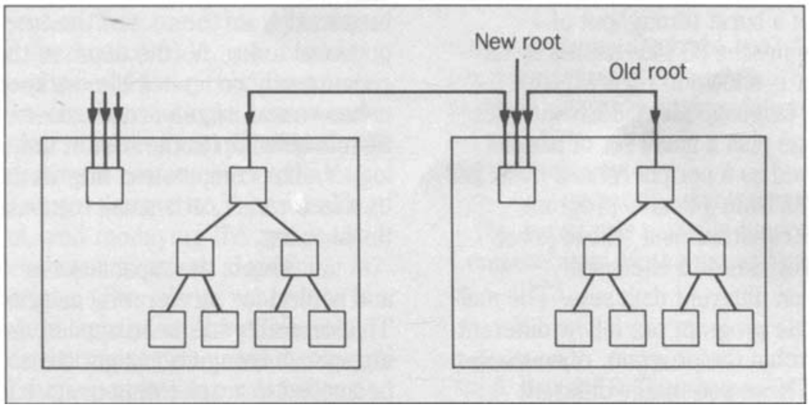


Figure 10. Expansion scheme for large tree machines.

that grows quadratically with the number of sensors. By a similar analysis, sensor contact formation has complexity that grows linearly with the number of sensors, and quadratically with the number of detections at a sensor. Thus high traffic conditions that cause increases in detections per sensor place particular demands on sensor contact formation.

Prior to the advent of scalable parallel computers, the only feasible approach to achieving real-time execution was to prune the comparisons via heuristics and knowledge. This led to procedures that indeed ran in real-time, but possibly missed many relationships and whose acceleration via heuristics was not directly transferable to other operational scenarios.

The ASPEN Parallel Processor. Given that one needs to accelerate contact formation via parallel processing, one must then select among a multitude of parallel processing machines. We are studying the applicability of the ASPEN parallel processor,[8] which is a tree machine. Tree machines are known to be applicable to accelerating pattern matching computations.[9] Since the computational core of contact formation involves an all-pairs comparison or match, ASPEN is an appropriate choice for accelerating contact formation. Tree machines have the additional advantage that they lead to compact, high-density processors.

ASPEN is a medium-grained parallel computer architecture, scalable to thousands of processing elements (PEs). It is currently in the prototype stage of development at AT&T Bell Laboratories, and comprises 127 PEs. Each PE comprises an 8-MFLOP AT&T DSP32 programmable digital signal processor, 64 Kbytes of local memory, and a communications processor. These PEs are then interconnected to form a complete binary tree. The expansion scheme for implementing large tree machines is due to Leiserson[10] and is shown in Figure 10. Each board-level module comprises a subtree plus an expansion PE. Two identical boards are then connected as shown, yielding a module with a subtree that is twice as large, plus an expansion PE. This process is then repeated to scale the machine to the desired size. The 8 by 13 inch ASPEN board shown in Figure 11(a) comprises 8 PEs. A 127-PE prototype comprises 16 such identical boards, takes up 1.5 cubic feet, and is capable of a burst throughput of 1 GFLOP. A workstation comprising 63 PEs hosted by an AT&T PC6300+ computer is shown in Figure 11(b).

The programming language for ASPEN includes the C programming language plus a small set of parallel constructs. ASPEN is viewed as a peripheral to a host, and is accessed via function calls from a host C program.

The major parallel construct is a "sliced procedure," in which identical programs are executed simultaneously in each PE on different data sets. The multiple executions of this single program can follow different instruction streams (still within the program, of course), depending upon the data. These potentially different instruction streams are forced to converge at the completion of the sliced procedure. This concept can be described as a single-program multiple-data (SPMD) machine. For adherents of the SIMD and MIMD[11] paradigms for parallel computation, an SPMD can be alternately viewed as a coarse-grain SIMD or a data-driven MIMD.

A *sliced procedure* is invoked as a function call from a host C program. In support of the sliced procedures, there are several global communication instructions, again invoked from the host C program. A `Broadcast` instruction transmits data from the host to all PEs. A `Resolve` instruction distinguishes the PE in the tree with the minimum value of some variable. A `Report` instruction, typically following a `Resolve`, transmits data from the distinguished PE to the host. The `Enable` and `Disable` instructions determine which PEs will execute a sliced procedure.

Parallel Connected-Components Procedure. Pattern matching involves comparing an unknown pattern against a known reference library and classifying the unknown as that reference pattern that it best matches. This is the computational core of contact formation, which involves comparing all objects with each other.

In the case in which the reference set can be enumerated and effectively listed, Bentley and Kung[9] describe how to efficiently exploit the parallelism in a binary tree with N PEs as follows: Distribute the reference patterns among the PEs. The unknown pattern is broadcast from the root of the tree to all PEs in time proportional to $\log_2 N$ (the depth of the tree). Each PE concurrently computes the distance scores between the unknown and its reference patterns. The best score is then bubbled up to the root in time again proportional to $\log_2 N$. The computation time is thus accelerated linearly by a factor of N, at a small communications cost proportional to $\log_2 N$.

This is the capability that makes ASPEN useful and natural for accelerating pattern recognition algorithms. This capability has been applied, for example, to accelerating speech recognition algorithms. The same capability can be applied to accelerating contact formation.

The Parallel Algorithm. A parallel algorithm for connected components clustering is presented that has several important attributes. It is incremental, in that if 1000 objects have been clustered, then the 1001st is processed without having to repeat any of the previous work. It includes update of the relationship graph (object comparison), as well as the connected components clustering (object clustering). The execution time per object remains essentially the same as the number of objects increases, so long as the number of PEs in ASPEN is scaled proportionally. This is very important from the perspective of systems engineering, making the execution time of large systems predictable. In addition, the source code at the host and PE levels also remains the same as problem size increases.

Panel 1 contains a pseudo-code program for parallel-connected components on ASPEN. The example given in the pseudo-code is for clustering sensor contacts into field contacts. It is assumed for the sake of exposi-

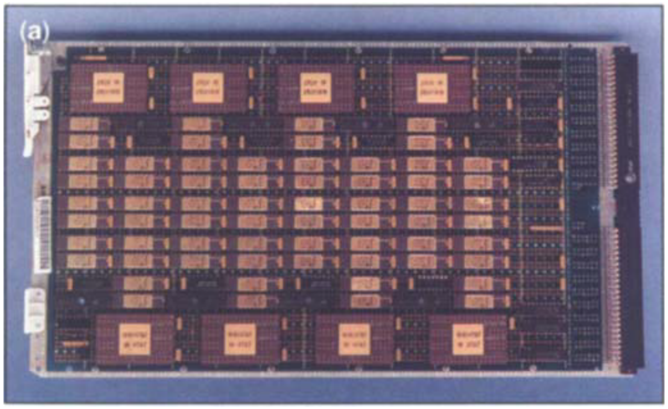
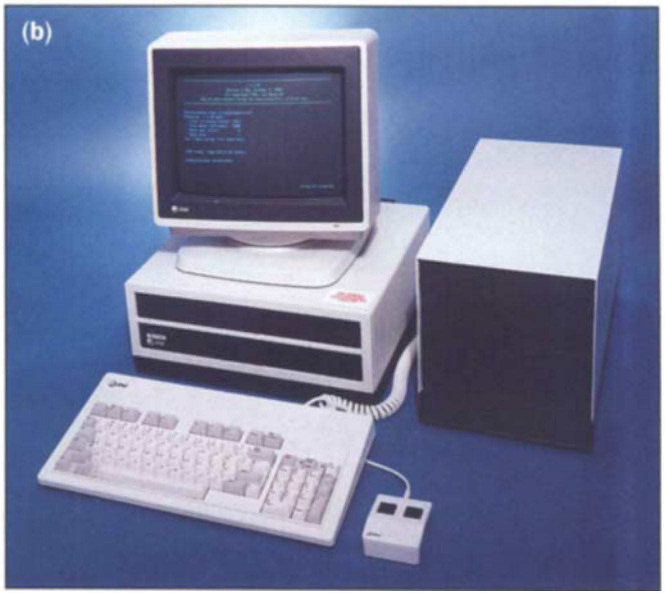



Figure 11. (a) ASPEN board with eight PEs. (b) ASPEN workstation with 63 PEs hosted by an AT&T PC6300+ computer.

tion that a single test is applied to pairs of sensor contacts, producing a single yes/no relationship. The algorithm also assumes virtual PEs (i.e., that objects are allocated, transparent to the source code, in a round-robin manner among the physical PEs). This construct allows the programmer to consider a single sensor contact per virtual PE. As described in Panel 1, steps in the algorithm include broadcasting a new sensor contact to all PEs, comparing that sensor contact to each "resident" sensor contact in parallel, and modifying the assigned field contact tags for the sensor contacts, depending on the results of the comparison. The merge procedure in step 7 of the pseudo-code is related to the choice of clustering procedure. For the connected components procedures, if two connected components are related to a common object, then they are merged into a single connected component. A different clustering procedure would have a different merge procedure. For example, a biconnected components procedure[12] would insist that two field contacts have at least two sensor contacts in common before merging them.

Conclusions

Signal interpretation is a rich and challenging problem that calls for the integration of various techniques and technologies. Various knowledge representation techniques, including object hierarchies and rule bases, are appropriate for different aspects of the problem. Similarly, various knowledge utilization procedures, including clustering algorithms or deductive retrieval procedures, are appropriate. Integration of parallel processing technology provides real-time processing capability. AT&T Bell Laboratories' current contact processing systems illustrate how AI and other technologies can be integrated to address the challenging real-world problem of signal interpretation.

Acknowledgment

The parallel clustering algorithm work described in this paper was partially funded by DARPA's Applied and Computational Mathematics Program, Contract N00039-86-C-0173.

References

1. A. Barr and E. Feigenbaum, *The Handbook of Artificial Intelligence*, Vol.2, William Kaufman, Los Altos, Calif., 1982.
2. E. Charniak and D. McDermott, *Introduction to Artificial Intelligence*, Addison-Wesley, Reading, Mass., 1985.
3. D. Moon and S. Keene, "New Flavors," *Proceedings*, Association for Computing Machines Object-Oriented Programming Systems, Languages, and Applications Conference, 1986.
4. M. R. Genesereth and N. J. Nilsson, *Logical Foundations of Artificial Intelligence*, Morgan Kaufmann, Los Altos, Calif., 1987.
5. P. R. Cohen, *Heuristic Reasoning About Uncertainty: An Artificial Intelligence Approach*, Morgan Kaufmann, Los Altos, Calif., 1985.
6. R. J. Brachman, "The Basics of Knowledge Representation and Reasoning," *AT&T Technical Journal*, Vol. 67, No. 1, January/February 1988, pp. 7-24.
7. M. Stefik and D. G. Bobrow, "Object-Oriented Programming: Themes and Variations," *AI Magazine*, Vol. 6, No. 4, Winter 1986.
8. A. L. Gorin and R. R. Shively, "The ASPEN Parallel Computer, Speech Recognition and Parallel Dynamic Programming," *Proceedings*, International Conference on Acoustics, Speech, and Signal Processing, April 1987, pp. 976-979.
9. J. Bentley and H. T. Kung, "A Tree Machine for Searching Problems," Technical report, Department of Computer Science, Carnegie-Mellon University, September 1979.

Panel 1—Kuzmak

```
Convention:
Host variables are lower case
Sliced variables are ALL CAPS

1. Get new_sensor_contact

2. Broadcast the new_sensor_contact from the host to NEW_SENSOR_CONTACT in the virtual PEs.

3. Concurrently, in all virtual PEs, compare NEW_SENSOR_CONTACT to the resident
   SENSOR_CONTACT and set the Boolean variable RELATED according to the results of
   that comparison.

4. Enable all virtual PEs whose SENSOR_CONTACT is RELATED to the
   NEW_SENSOR_CONTACT and disable all others.

5. If there are no RELATED SENSOR_CONTACTS (i.e., if all virtual PEs are disabled), then:
   - Allocate a new virtual PE for the new_sensor_contact, storing it there as a
     SENSOR_CONTACT and assigning to it an as-yet unused FIELD_CONTACT tag.

6. If all RELATED SENSOR_CONTACTS have the same FIELD_CONTACT tag, then:
   - Allocate a new virtual PE for the new_sensor_contact, storing it there as a
     SENSOR_CONTACT and assigning to it the FIELD_CONTACT tag
     of the RELATED SENSOR_CONTACTS.

7. If the RELATED SENSOR_CONTACTS have different FIELD_CONTACT tags, then:
   - Allocate a new virtual PE for the new_sensor_contact
   - For each of the RELATED SENSOR_CONTACTS, enable all other SENSOR_CONTACTS with the
     same FIELD_CONTACT tag.
   - Merge all enabled SENSOR_CONTACTS by assigning them the same LD_CONTACT tag.
```

10. C. E. Leiserson, "Area-Efficient VLSI Computation," Ph.D. Thesis, Carnegie-Mellon University, 1981.
11. M. J. Flynn, "Very High-Speed Computing Systems," *Proceedings of the IEEE*, Vol. 54, No. 12, December 1966, pp. 1901-1909.
12. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, Mass., 1974.

Biographies (continued)

an M.A. in psychology and an M.S. in computer science and will shortly receive a Ph.D. in psychology, all from the University of Delaware. He joined AT&T in 1986. Mr. Gorin is a distinguished member of technical staff. His research interests are parallel algorithms and architectures for pattern recognition. He received a B.S. and M.A. in mathematics from the State University of New York at Stony Brook and a Ph.D. in mathematics from the City University of New York. He joined AT&T in 1983. Mr. Milich is a member of technical staff, responsible for software design and development of knowledge-based systems for signal interpretation. He received a B.S. in information systems management and an M.S. in computer science from the State University of New York at Buffalo. He joined AT&T in 1983. Mr. Shoenfelt is supervisor of the Knowledge-Based Systems Design group, which applies signal interpretation and knowledge-based system technology to a wide range of military problems. He received a B.S. from Union College and an M.S. and Ph.D. from North Carolina State University, all in electrical engineering. He joined AT&T in 1970.

(Manuscript received November 3, 1987)