Distributed Symbolic-SubSymbolic Agent Architecture for Configuring Power Network Faults

R. Khosla **T. Dillon**
Expert and Intelligent Systems Laboratory,
Applied Computing Research Institute, and
Department of Computer Science and Computer Engg.,
La Trobe University, Melbourne,
Victoria, Australia
e-mail: khosla@latcs1.lat.oz.au
Tel: +61.3.4793034, Fax: +61.3.4704915

Extended Abstract

Intelligent Agents among other aspects represent task abstraction. Tasks and methods (used to accomplish the tasks) can be considered as mediating concepts in problem solving. The symbolic and connectionist methods because of their different philosophical, cognitive and computational underpinnings impose their own constraints on the quality (e.g time, resources, no. of levels of abstraction/solution hierarchy, reliability, etc.) of task accomplishment. Further, these constraints can lead to task generation depending upon which global constraints have been violated by a certain method, and the degree of tolerance of the violated constraints. A proper integration of symbolic and connectionist methods for task accomplishment can thus improve the quality of task accomplishment and restrict task generation. More so, such a integration can provide more flexibility to an agent in terms of multiplicity of methods for accomplishing a task.

In this direction our work involves development of multi-agent symbolic-subsymbolic architecture at the task structure level, computational level, and program level for complex knowledge intensive domains.

The architecture at the three different levels is based on pragmatic constraints like cognitive compatibility (in terms of information processing),neurobiological considerations, forms of knowledge, problem complexity, search time, temporal reasoning, learning, incorrect and incomplete information, reliability, generalization, distributed processing, global and local parallelism, global and local competition, cooperation, knowledge sharing/reuse, inheritance, scalability, and cost effectiveness.

The task structure level architecture is a product of the tasks, constraints (i.e. global and local constraints) on the tasks and the methods used to accomplish the tasks. The tasks are defined within five phases of information processing, namely, preprocessing phase, decomposition phase, control phase, decision phase , and postprocessing phase. The division of responsibility between symbolic and connectionist methods is based on the satisfaction of various task constraints.

The distributed computational level architecture is derived through integration of task structure level architecture with an object-oriented model and an unix operating system process model. The computational level architecture consists of preprocessing and decomposition agents at the global level and control, decision, and postprocessing agents at the local level. At the computational level we determine the knowledge content of the symbolic-connectionist agents. The symbolic-connectionist knowledge content includes among other aspects the planning knowledge, belief knowledge, a *priori* learning knowledge (for Artificial Neural Networks), execution knowledge, communication knowledge, and validation (in terms of time and context) knowledge for preprocessing, decomposition, control, decision, and postprocessing agents.

The knowledge content in general represents three forms of knowledge, i.e. symbolic and informal (conceptual and heuristical), sub-symbolic (subconceptual), and symbolic and formal (logic, mathematical models) knowledge. The integration of the task structure level architecture with an object-oriented model and unix operating system process model also enables us to satisfy constraints like distributed processing, inheritance, knowledge sharing and reuse. The computational architecture has been successfully used for developing a real-time distribu ted, alarm processing system for configuring network faults in a power system control centre with encouraging performance results. The objectives of the alarm processing system are a) isolate the event, b) isolate the cause of the event, and c) isolate the cause of the event in four seconds for simple faults and maximum of thirty seconds for severe faults. Real time alarm data from the power system control centre has been used in the alarm processing system.