An Intelligent Approach to Welding Robot Selection

J. Milano, S.D. Mauk, L. Flitter, and R. Morris

In a shipyard where multiple stationary and mobile workcells are employed in the fabrication of components of complex sub-assemblies, efficient operation requires an intelligent method of scheduling jobs and selecting workcells based on optimum throughput and cost. The achievement of this global solution requires the successful organization of resource availability, process requirements, and process constraints.

The Off-line Planner (OLP) of the Programmable Automated Weld System (PAWS) is capable of advanced modeling of weld processes and environments as well as the generation of complete weld procedures. These capabilities involve the integration of advanced Computer Aided Design (CAD), path planning, and obstacle detection and avoidance techniques as well as the synthesis of complex design and process information. These existing capabilities provide the basis of the functionality required for the successful implementation of an intelligent weld robot selector and material flow planner.

Current efforts are focused on robot selection via the dynamic routing of components to the appropriate work cells. It is proposed that this problem is a variant of the "Traveling Salesman Problem" (TSP) that has been proven to belong to a larger set of optimization problems termed nondeterministic polynomial complete (NP complete). In this paper, a heuristic approach utilizing recurrent neural networks is explored as a rapid means of producing a near optimal, if not optimal, weld robot selection.

Keywords
Hopfield networks, neural networks, optimization, path planning, robot, welding

1. Introduction

IT has long been accepted that cost effectiveness in the application of robot manipulators to heavy manufacturing processes is dependent on the ability to provide autonomous systems with timely process planning information. The advent of computer integrated manufacturing (CIM) technology is a direct attempt to provide process, environment, and part simulation in sufficient detail to allow for the off-line programming of autonomous systems. Current state of the art CIM systems support the transition of information from CAD-based design systems to robotic manipulator programs in a wide variety of manufacturing applications.

A current focus of CIM technology is on the definition of process modeling for a given set of operations employing a specific manipulator, or set of manipulators, operating independently. No capability currently exists for expanding the modeling capability enterprise within the total manufacturing environment. Such a capability would provide optimization for the overall manufacturing environment from the standpoint of resource utilization. Additionally, the provision of this capability within a CIM environment would provide opportunities to dynamically perform iterative design for manufacturability analysis. In the long term, this approach will be a valuable tool in evaluating in-place workcells and for design and implementation of new workcells.

J. Milano, L. Flitter, and **R. Morris,** Naval Surface Warfare Center, Carderock Division, Code 2033, Bethesda, MD, 20084-5000; and **S.D. Mauk,** SDM Consulting Inc., P.O. Box 359, Upper Marlboro, MD 20772.

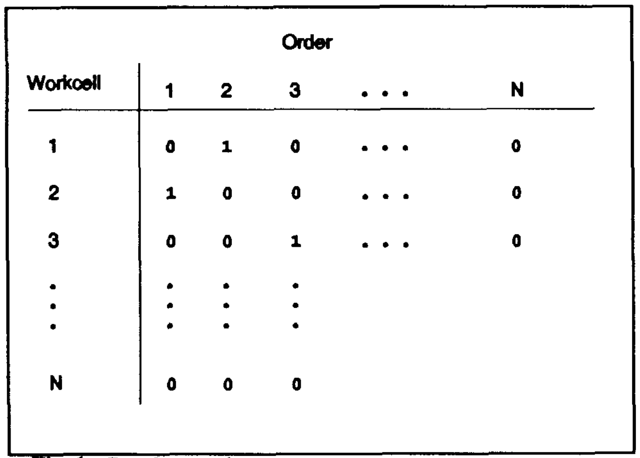
2. Background

In a modern shipyard environment, large subassemblies comprised of many smaller hull and structural components are produced in semiautomated workcells. A major impediment to increased automation at the workcell level is the inability to manage overall material flow through the workcells. Autonomous systems are routinely used on only a small percentage of the welding tasks. If an optimal schedule of weld task execution could be rapidly devised for each subassembly, robot manipulators specifically selected for each task could be appropriately dispatched. This article proposes a conceptual method by which an optimal robot selection and execution schedule can be automatically generated from enhanced CAD data.

The programmable automated welding system (PAWS) is an integrated system for performing automated welding.[1] This system was initially developed for shipyard applications and incorporates subsystems for planning and real-time control. The PAWS off-line planning (OLP) system provides a CAD database interface for the acquisition of part specifications.[2] The OLP represents these part specifications as solid models and overlays process and path information relevant to a given manufacturing process (i.e., welding). The result is an enhanced CAD representation complete with weld process information.

The OLP is capable of simulating robot motion and process operations including collision detection and collision avoidance. Currently, the OLP focuses on planning for single-part manufacturing for a given robot manipulator. However, the OLP can simulate any manipulator for which a suitable kinematic solution is available. This article proposes a concept to augment the PAWS OLP, adding the capability to perform intelligent selection of robot manipulator and manufacturing sequencing.

When each weld task is considered in an isolated sense, the selection of a robot manipulator for fabrication is a trivial task.



	Order				
Workcell	1	2	3	. . .	N
1	0	1	0	. . .	0
2	1	0	0	. . .	0
3	0	0	1	. . .	0
⋮	⋮	⋮	⋮		
N	0	0	0		

Fig. 1 Results matrix.

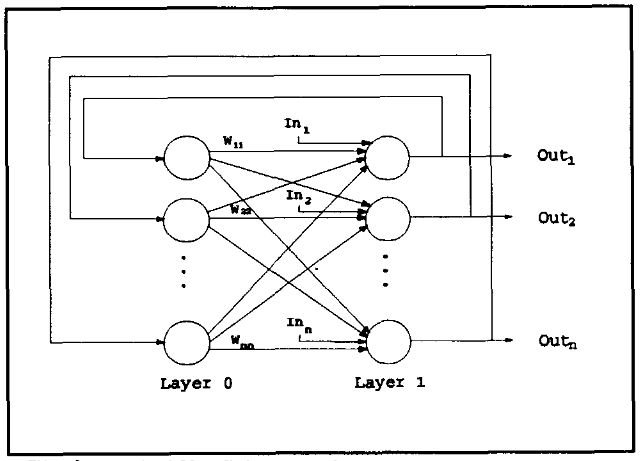


Fig. 2 Typical Hopfield network.

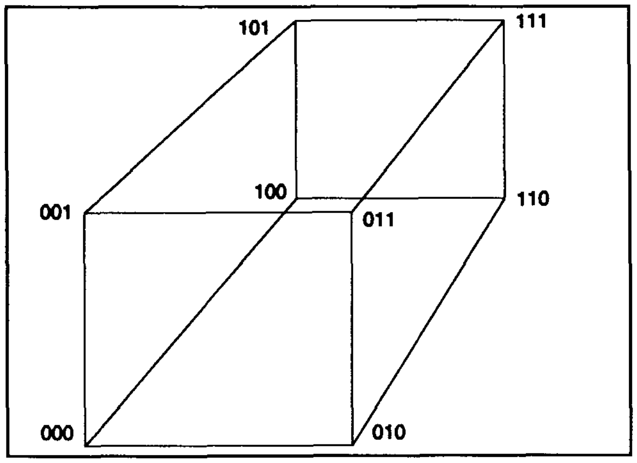


Fig. 3 Energy lattice.

A straightforward comparison of the ability of each available manipulator to kinematically satisfy reach, clearance, and position constraints would yield a satisfactory result. The result, however, does not address optimal planning for the overall fabrication process. When taken in the context of manufacturing entire subassemblies (consisting of potentially thousands of weld tasks), a task scheduling component must be introduced to (1) assign each job to the best suited workcell and (2) assign all jobs so as to achieve level loading among workcells.

3. Approach

It is proposed that the scheduling of autonomous shipyard welding tasks can be represented as a modified variant of the "traveling salesman problem" (TSP). The TSP is a subset of a general class of optimization problems known as nondeterministic polynomial complete (NP complete).[3] The NP complete problems have no known method of solution other than exhaustively searching the solution space. An exhaustive search is not computationally practical for most applications; thus, heuristic techniques have been applied, yielding near optimal, if not optimal, solutions. In this article, a heuristic approach using recurrent neural networks[4] is explored as a rapid means of producing a near optimal weld robot selection.

The TSP demands that one find the best order among the n cities to be visited that minimizes the total distance traveled. Here, a cost function, which is dependent on the order of a finite number of objects, has to be minimized. In this analogy, each autonomous cell would be viewed as a "stop" along the path of subassembly construction. The complexity of the requisite scheduling analysis would increase on the order of $(n-1)!$, where n = number of workcells. Given that any optimum solution would require each subassembly to visit a workcell only once, a matrix similar to the one depicted in Fig. 1 could be derived. Taking the TSP analogy further, the traditional TSP cost function (the minimization of total distance traveled) would be replaced by a cost function defined as a weighted, linear combination of system constraints such as cell setup time, order of assembly, percent flat position welds, percent utilization of manipulator, and percent manual effort required to complete. Although this list of cost parameters is by no means complete, it serves to sufficiently illustrate the application of TSP to weld task scheduling.

The application of a connectionist model to provide rapid solutions to the TSP problem would be desirable. However, the large number of potential solutions makes the training of more traditional neural networks, such as standard back propagation, a formidable task. Fortunately, an intelligent algorithm for solving TSPs already exists in the form of Hopfield nets (Fig. 2). Hopfield nets are inherently recurrent because of the feedback connection of network outputs to input vectors. The recurrent network ability to function as an associative memory[5] makes possible the selection of an optimal solution given incomplete data.

A geometric representation of the solution space can be used to illustrate the behavior of a Hopfield network.[6] For example, a three neuron system can be represented by a cube in three-dimensional space, as depicted by Fig. 3. Each of the eight vertices corresponds to one of the eight possible system states and is labeled by a three-bit binary number, where each bit corresponds to a neuron output. Generalizing this concept, a network having n neurons will have 2^n discrete states associated with an *n*-dimensional hypercube. Stimulating the network with an input vector sets the state of the network, corresponding to a vertex on the hypercube. The state of the network then shifts from vertex to vertex until stability is achieved; that is, until an energy minima is reached. The stable vertex is defined as a function of the networks connection weights, the current input, and the threshold value. If the information in this vector is incomplete or partially incorrect, the system stabilizes to the vertex closest to the one desired. This ensures that *an* optimal, if not *the* optimal, solution will be reached.

Unlike nonrecurrent neural networks, recurrent networks are dynamically responsive. That is, after applying a new input, the output is recalculated and fed back into the network in an iterative manner. For a stable network, each iteration diminishes the change in the output until a steady state is achieved. Unstable networks, however, iterate endlessly and are thus treated as chaotic systems. Recurrent networks are stable if the weight matrix is symmetrical with zeros on the main diagonal.[7] If one denotes the elements of the weight matrix by $\eta_{i\alpha}$, where $i = 1, ..., n$ stands for the workcell and $\alpha = 1, ..., n$ indicates the workcell position in the manufacturing sequence, such a network would have an associated Liapunov function of the form:

$$\varepsilon = (1/2)\left[\sum_{i,k}^{i\neq k}\sum_{\alpha,\beta}^{\alpha\neq\beta} \omega_{i\alpha,k\beta}\,\eta_{i\alpha}\,\eta_{k\beta}\right] \qquad [1]$$

where ε is network energy; ω is synaptic connections; and η is the weight matrix.

The expression of the cost function to be minimized in the form of the Liapunov function ensures network stability, as any change in the state of a neuron will either reduce the network energy or maintain its current value.

The robot selection process must be mapped onto the computational network by expressing the associated cost function in the form of the Liapunov function. In this application, the energy function must satisfy two requirements: (1) it must be low for only those solutions that produce a single neuron activation (each workcell can only be visited once) and (2) it must favor solutions that satisfy system constraints. The first requirement is satisfied by an energy function of the form:

$$\varepsilon_1 = (A/2)\sum_{i,\alpha,\beta}^{\alpha\neq\beta} \eta_{i\alpha}\,\eta_{i\beta} + (B/2)\sum_{i,k,\alpha}^{i\neq k} \eta_{i\alpha}\,\eta_{k\alpha} \qquad [2]$$

where *A* and *B* are Lagrange multipliers.

The resulting set of rules is as follows. The first term is zero if, and only if, each workcell is visited only once. The second term is zero if, and only if, that position in the sequence is not occupied by another workcell.

System constraints are expressed as a weighted linear combination in the following fashion:

$$\zeta_{i\alpha} = \sum_{j} w_{o\dot{g}}\,\gamma_{io\dot{g}} \qquad [3]$$

where $\zeta_{i\alpha}$ is the system specific cost function for workcell *i* at position α; $\gamma_{io\dot{g}}$ is the system specific constraints for workcell *i* at position α; and $w_{o\dot{g}}$ is the weighting coefficients for position α.

This allows the second requirement, minimizing system specific constraints, to be satisfied by adding an additional term to the energy function in the following fashion:

$$\varepsilon_2 = \sum_{i,\alpha} \zeta_{i\alpha}\,\eta_{i\alpha} \qquad [4]$$

Combining terms from Eq 2 and 4 yields:

$$\varepsilon = \varepsilon_1 + \varepsilon_2 \qquad [5]$$

This approach can be extended to a scenario where multiple parts are involved and level loading across workcells is desired. Equation 6 illustrates this relationship for two parts:

$$\varepsilon_{\text{total}} = \vee\, [C(|\varepsilon_I - \varepsilon_{II}|) + D(\varepsilon_I + \varepsilon_{II})] \qquad [6]$$

where ε_I and ε_{II} are energy functions for each part; and *C* and *D* are constants.

4. Conclusion

Through the application of Hopfield networks to a problem of the TSP form effective weld task scheduling can be accomplished. The requisite data for the calculation of weld task scheduling are provided as a result of the off-line weld job planning process. The data are resident in an augmented CAD format and can be readily assembled into a weighted cost equation for each subassembly to be constructed. Further augmentation of the cost equation to include more manufacturing specific parameters will provide a model suitable for the scheduling of tasks in an operational environment.

Alternative representations of the expression of the manufacturing cost function as a linear, weighted sum must be explored. Nonlinear dependent representations will certainly be required to represent many manufacturing applications. Parallel efforts to better define this cost function are appropriate, but these efforts remain beyond the scope of this article.

References

1. M.D. Kline, Programmable Automated Welding System (PAWS): System Overview, *3rd Int. Welding Research Conf.*, Gatlinberg, TN, ASM International, 1992
2. J.S. Hemmerle, M. Terk, E.L. Gürsöz, F.B. Prinz, and T.E. Doyle, Planning in PAWS: A New Approach to Robotic Welding, *3rd Int. Welding Research Conf.*, Gatlinberg, TN, ASM International, 1992
3. M.R. Garey and D.S. Johnson, *Computers and Intractability*, W.H. Freeman, 1979

4. J.J. Hopfield and D.W. Tank, Neural Computation of Decisions in Optimization Problems, *Biological Cybernetics*, Vol 52, 1985, p 141-152

5. J.J. Hopfield, Neurons with Graded Response Have Collective Computational Properties Like Those of Two-State Neurons, *Proc. Nat. Acad. Sci.*, Vol 81, 1984, p 3088-3092

6. P.D. Wasserman, *Neural Computing: Theory and Practice*, Van Nostrand Reinhold, 1989

7. M.A. Cohen and S.G. Grossberg, Absolute Stability of Global Pattern Formation and Parallel Memory, Storage by Competitive Neural Networks, *IEEE Trans. Systems, Man, Cybernetics*, Vol 13, 1983, p 815-826

Additional References

- B. Müller and J. Reinhardt, *Neural Networks—An Introduction*, Springer-Verlag, 1990
- D.E. Van den Bout and T.K. Miller, A Traveling Salesman Objective Function that Works, *Proc. IEEE Int. Conf. Neural Networks*, San Diego, Vol 2, 1988, p 299-304