Optimal Task Allocation in a Multirobot Environment Using Capability Indices

1st International Conference on Advances in Computing, Chikhli, India, 21-22 February 2008

B.B.Choudhury and B.B. Biswal

Optimal Task Allocation in a Multirobot Environment Using Capability Indices

B. B. Choudhury
Department of Mechanical Engineering
National Institute of Technology
Rourkela-769008, INDIA
+91-9437166989
bbcnit@gmail.com

B. B. Biswal
Department of Mechanical Engineering
National Institute of Technology
Rourkela-769008, INDIA
+91-9437158476
bibhuti.biswal@gmail.com

ABSTRACT
Multirobot systems (MRS) hold the promise of improved performance and increased fault tolerance for large-scale problems. Multirobot coordination, however, is a complex problem. One of the most important aspects in the design of multi-robot systems is the allocation of tasks among the robots in a productive and efficient manner. An empirical study is described in the present paper for task allocation strategies. In general, optimal solutions can be found through an exhaustive search, but because there are $n \times m$ ways in which m tasks can be assigned to n robots, an exhaustive search is often not possible. Task allocation methodologies must ensure that not only the global mission is achieved, but also the tasks are well distributed among the robots. An effective task allocation approach considers the available resources, the capabilities of the deployable robots, and then it appropriately allocates the tasks the candidate robots. This paper presents such task allocation methodologies for multi-robot systems by considering their capability in terms of time and space.

Keywords
Allocation model, assignment heuristic, allocation cost, load deviation ratio, multirobot, task allocation

1. INTRODUCTION
The study of multirobot system (MRS) has received increased attention in the recent years. Continually improving technology has made the deployment of MRS consisting of increasingly larger number of robots possible. It is obvious that multiple robots will be superior to a single robot in achieving a given task. Potential advantages of MRS over single robot systems (SRS) include reduction of total system cost by employing multiple simple and cheap robots as opposed to a single, complex and expensive robots. The inherent complexity of certain task environment may require the use of multiple robots as the demand for capability is quite substantial to be met by a single robot. Multiple robots are assumed to increase system robustness by taking advantage of inherent parallelism and redundancy.

Multirobot teamwork is a complex problem consisting of task division, task allocation, coordination, and communication. The most significant concept in multi-robot systems is cooperation. The cooperation of robots in a group can be classified into two categories of implicit cooperation and explicit cooperation. In the implicit cooperation case each robot performs individual tasks, while the collection of these tasks is toward a unified mission. The explicit cooperation is the case where robots in a team work synchronously with respect to time or space in order to achieve a goal. Regardless of the type of cooperation, the goal of the team must be transformed in to tasks to be allocated to the individual robots. In this paper, an attempt is made to empirically derive some guidelines for selecting task allocation strategies for multi-robot systems with implicit cooperation. The explored strategies are individualistic in that they do not involve explicit cooperation and negotiation among the robots. However, they are a part of a large class approaches that produce coherent and efficient cooperative behavior. The approach presented in this paper can be advantageously used in real-world problems. The present work is aimed at proposing a methodology to allocate tasks to available multiple robots based on their capacity, availability and allocation cost.

The allocation model (AM) is equivalent to a two-dimensional multi-type bin packing problem (2DMBP). Considering each robot as a two-dimensional bin, and each workstation a two-dimensional object to be packed, the model can be viewed as assigning objects into an optimal set of bins such that both resource demands of each object are satisfied and neither of the capacity constraints of each selected bin is violated. Historical research in bin packing has focused on both the one-dimensional single-type bin packing problem (1DSBP) and its two dimensional extension-2DSBP [1], [2]. Both 1DSBP and 2DSBP have been proven NP-hard [3], [4]. Numerous investigators have examined the performance analysis of approximation algorithms designed for a number of 2DSBP variants [4,5,6,7]. Reported algorithms may be categorized as either on-line or off-line. Both types of algorithms are practical in real world applications. In this paper, the focus is on the development and implementation of an optimization algorithm for solving AM. Specifically, the objective of this work is to develop a solution algorithm that can be used to solve problems of a practical size within acceptable computational times. The characteristics of AM warrant the development of an off-line algorithm involving vector-packing. Although the algorithm described here is in the context of robotics, it is general and applicable to any real-world application.

2. ALLOCATION MODEL FORMULATION

The present research problem explicitly addresses robots of different types with various service time and space capacities. The assignment model (AM) seeks an optimal selection of robots to serve all given work stations such that each work station's resource demands are satisfied, no robot capacity constraints are violated, and the total system cost is minimized. A mathematical model along with its solution algorithm is presented for allocation of robots to the workstations which is efficient and may serve as a planning tool. The model is formulated as a pure 0-1 mathematical program. The key parameters for the model can be categorized as geometrical, kinematic, dynamic, power and noise, and thermal. The two most important factors when assigning work stations to robots are the geometrical work envelope and the kinematic machine cycle time. The work envelope for a typical robot is represented by a diameter of a circle. However, for our model, it is not required that the work envelope be a complete circle. The time requirement of any workstation depends upon its relative distance from the robot. In addition, the space requirement of a workstation also depends upon its relative location. If the workstation is assigned to location nearer to the robot its space requirement is smaller than what is required if assigned at location one. In contrast, the time requirement of a workstation assigned at location farther to robot is smaller than the time requirement associated with location which is nearer from robot because the latter incurs a longer travel time. Thus, there exists a trade-off between the space requirement and machine cycle time requirement. In fact, both requirements are a function of the workstation's relative position from the robot. Our primary objective is to minimize the total robot acquisition costs while satisfying workstation resource demands. In order to make AM computationally tractable, we assume that all workstations are placed at the most remote location within the work envelope. This assumption decouples the interaction between space and time by allowing the resource requirements of a given workstation to be constant. Without this assumption, the model complexity is significantly increased. This trade-off between the number of robots required to serve a given set of workstations and the time required to serve a workstation could be considered by iteratively solving AM. The formulation of the AM is as follows. A set of robot types indexed by K = {1, 2, …, k}, is considered where each robot type is characterized by its time and space capacity. Specifically, space is measured in terms of the work envelope's swept area. The swept area is the total number of degrees around the central vertical axis that is within reach of the robot arm. All given workstations are indexed by I = {1, 2, …, n}.

Table 1. Notation for AM

Notation	Definition
k	Number of robot types
K	Robot type index set, K = {1, 2, ..., k}
n	Number of workstations
I	Workstation index set, I = {1, 2, ..., n}
m_k	Maximum number of type k robots $\forall k \in K$
J_k	Robot index set for type k robots, J_k= {1,2,…, m_k}, $\forall k \in K$
t_{ik}	Normalized time requirement of workstation i when served by a type k robot, $\forall i \in I, k \in K$
s_{ik}	Normalized space requirement of workstation i when served by a type k robot, $\forall i \in I, k \in K$
f_k	Fixed charge incurred if `a type k robot is purchased, $\forall k \in K$

Each workstation i demands a known amount of time and space when served by robot type k, denoted by t_{ik} and s_{ik} respectively. In addition, for a given set of n workstations, let m_k denote the maximum number of robots of type k necessary to serve all workstations assuming only robots of type k are available. Further, let J_k = {1, 2, ..., m_k} denote the index set for type k robots. It is assumed that a fixed-charge, f_k, will be incurred if the j^{th} robot of type k is employed. For easy reference, all useful notations for AM are summarized in Table 1. A decision variable, x_{ijk}, is defined as:

$$x_{ijk} = \begin{cases} 1 \text{ if workstation } i \text{ is assigned to robot } j \text{ of type } k \\ 0 \quad \text{otherwise} \end{cases}$$

With no loss of generality, the time and space requirements for each workstation (i.e., t_{ik}, and s_{ik} , respectively) can be normalized by dividing the robot resource capacities into the corresponding workstation resource demands. This macro planning model does not consider variable costs and the solution algorithm developed in this paper is general. The robot selection and assignment (RSA) can be written as equation (1) through equation (6):

$$\text{(RSA) MIN Z} = \left(\sum_{k \in K} f_k \left(\sum_{j \in J_k} y_{jk} \right) \right) \tag{1}$$

Such that

$$\sum_{k \in K} \sum_{j \in J_K} x_{ijk} = 1 \quad \forall i \in I \tag{2}$$

$$\sum_{i \in I} t_{ik} x_{ijk} = 1 \quad \forall k \in K, j \in j_k \tag{3}$$

$$\sum_{i \in I} s_{ik} x_{ijk} \le 1 \quad \forall k \in K, j \in j_k \tag{4}$$

$$x_{ijk} \le y_{jk} \quad \forall i \in I, k \in K, j \in J_k \tag{5}$$

$$x_{ijk}, y_{jk} \in \{0,1\} \quad \forall i \in I, k \in K, j \in J_k \tag{6}$$

Condition (2) ensures that each workstation i is assigned to exactly one robot. Conditions (3) and (4) ensure that workstations assigned to any robot will not violate the corresponding time and space constraints. The AM is a pure 0-1 integer program (IP). Therefore, it is impractical to directly solve AM by using any available IP code. In the present work an optimization algorithm based on heuristic covering all the necessary parameters is developed for solving the task assignment problem in a heterogeneous multirobot environment.

3. SOLUTION METHODOLOGY

The solution algorithm for AM uses a greedy heuristic-First Fit by Ordered Deviation (FFOD) to generate an initial feasible solution. The algorithm is used to search for the optimum. The heuristic provides an initial feasible solution which serves as an upper bound. This solution and its corresponding objective function value are then iteratively expanded and solved by using a decomposition procedure. At the end of each iteration, new groupings are generated [8]. The best or a few good groupings are appended to the problem. This iterative solution process continues to refine the objective function until no more new groupings with non-negative reduced costs can be generated [9,10]. If the final solution is all integers, then an optimal solution to the original AM problem has been found and the algorithm terminates.

4. ALGORITHM DETAILS

4.1 Assignment Heuristic (AH)

Given any workstation, three possibilities exist. The workstation can be time intensive, space intensive or neither. A heuristic is developed by examining these three cases, and the load balance on each candidate robot. The AH is based on the concept of allocation cost, which is computed as a function of the resource demands of each workstation and a robot's load balance. Let Δ_{jk} denote the load balance factor associated with the j^{th} robot of type *k*. That is, Δ_{jk} is defined as the difference between the total allocated (normalized) machine time and the total allocated (normalized) work space for robot *j* of type *k*. Let and where x_{ijk} is a 0-1 variable. Hence, Δ_{jk} can be expressed as follows

$$\Delta_{jk} = T_{jk} - S_{jk} \quad \forall k \in K, j \in j_k \tag{7}$$

If a robot's resource load is nearly balanced, then the load balance factor will be approximately zero. If the robot's load is time intensive, then $0 < \Delta_{jk} < 1$, and if the robot's load is space intensive, then $-1 < \Delta_{jk} < 0$. Hence, the further the resource load factor is away from zero, the greater the load imbalance is.In addition, let δ_{ijk} denote the adjusted demand when the i^{th} workstation is served by the j^{th} robot of type *k*. That is,

$$\delta_{ijk} = \begin{cases} \text{MAX}\{t_{ik}, s_{ik} - \Delta_{jk}\} \text{ if } \Delta_{jk} > 0 \\ \text{MAX}\{t_{ik} + \Delta_{jk}, s_{ik}\} \text{ if } \Delta_{jk} \le 0 \end{cases} \tag{8}$$

Since $0 < t_{ik} \le 1$, $0 < s_{ik} \le 1$, and $-1 < \Delta_{jk} < 1$, we know $0 < \delta_{ijk} \le 1$.

To illustrate how the adjusted demand is employed by AH, consider two robots of type *k*, say A and B. Assume that $\Delta_{Ak} = 0.4$ and $\Delta_{Bk} = -3$. Therefore, robot A is time intensive. In order to improve the load balance for robot A, we should prefer the assignment of a workstation which is space intensive (i.e., $t_{ik} < s_{ik}$) to those which are time intensive. By contrast, for robot B, the assignment of workstations which are time intensive should be given preference over workstations which are space intensive. An example is given below for illustration. Suppose the workstation to be assigned next is time intensive; that is, $t_{ik} = 0.3$ and $s_{ik} = 0.2$. Also, assume both robot A and B have enough remaining time and space capacities to serve this candidate workstation. The goal of our assignment heuristic is to balance the resource load on each robot. Since the candidate workstation is time intensive, it should be assigned to a robot which is space intensive. Plugging the given figures into equation (8), we have $\delta_{iAk} = 0.3$ and $\delta_{iBk} = 0.2$. These adjusted demands, i.e., δ_{ijk} contribute to the "allocation costs". In general, if the fixed cost of all robot types is equal, the workstation should be assigned to the robot which produces the smallest adjusted demand. Since, not all robots have equal fixed cost, the allocation cost, a_{ijk} incurred by the i^{th} workstation when served by the j^{th} robot of type *k* is the product of its adjusted demand and the fixed cost of the j^{th} robot. That is,

$$a_{ijk} = f_k * \delta_{ijk} \tag{9}$$

Since $0 < \delta_{ijk} \le 1$, we know that $0 < a_{ijk} \le f_k$. Thus, a_{ijk} reflects the adjusted proportion of the fixed cost that workstation i incurs when it is assigned to robot *j* of type *k*. The heuristic is used to produce a good feasible solution. For each robot type *k*, the heuristic calculates the load deviation ratios and sorts them into a nondecreasing order. These load deviation ratios indicate the balance between the time and space requirements of each workstation when served by each robot type *k*. Then, the AH is employed to assign workstations to robots based on the sorted load deviation ratios. Since AH is simple and efficient, it is rerun once more based on a nonincreasing order of load deviation ratios. Our computational results indicate that AH provides a very good feasible and optimal solution.

4.2 AM Example

Using realistic data, the following example is provided to highlight the solution process for a AM problem. While Table 2 summarizes major parameter values for four different robot types, Table 3 presents the normalized space and time requirements of fifteen workstations.

Robot-4 has a fixed charge of \$60,000, a swept area of 320°, a maximum reach of 1250 mm, and an average arm movement speed of 3.09 m/sec. Each entry in column two of Table 3 provides the diameter (*D*) of a circle encompassing the workstation. It is assumed that each workstation is placed at the most remote location within the work envelope. Therefore, the *D* associated with each workstation is in fact a chord to the work envelope. Knowing the value of *D* and the maximum reach (*R*) of a robot, we can derive the arc length subtended by a workstation, which is $R\theta$ *where* $\theta = 2\sin^{-1}(D/2R)$. Here, θ represents the workstation's space requirement in degrees. Using θ and the swept area (*S*), a workstation's normalized space requirement can be determined. Considering workstation one and robot type one, we have D = 1.0 meter, R = 1.25 meters, and $S = 320^\circ$. Using this data, we have $\theta = 47.15^\circ$ and thus $S_{11} = (47.15/320) = 0.147$. In contrast, the time requirement of a workstation can only be determined after a thorough motion study of robot. In this macro planning model, the time requirement for each workstation is estimated based on two major components: (1) robot arm travel time; (2) robot service time. Both components are normalized by the total available machine time, which in practice is defined by the time available during peak machine hours. Using the above data and the aforementioned optimization algorithm, the AM example is optimally solved. To proceed with the solution for allocation model, all the options of employing the available robot types are tried. The load balance factor Δ_{jk} and the allocation cost for each option are determined.

Table 2. Fixed costs and parameter values of the robots

	Robot-1	Robot-2	Robot-3	Robot-4
Specification	(Puma 560-c)	(Adept one XL)	Fanuc Arcmate Sr.R.J	Staubli RX 130B
DOF	6	4	6	6
Pay Load	4 kg	12 kg	10 kg	12 kg
Swept Area	320°	270°	300°	320°
Max. Reach	878 mm	800 mm	1529 mm	1250 mm
Max Speed	1.0 m/sec	1.2 m/sec	3.60 m/sec	3.09m /sec
Type	Jointed	Scara	Jointed	Jointed
Cost	\$35,000	\$19,500	\$56,400	\$60,000

Table 3. Normalized space and time requirements of workstations

Workstation		Normalized space requirement				Normalized time requirement			
No.(i)	Size(D)	R-1	R-2	R-3	R-4	R-1	R-2	R-3	R-4
		S_{i1}	S_{i2}	S_{i3}	S_{i4}	t_{i1}	t_{i2}	t_{i3}	t_{i4}
1	1.0	0.216	0.286	0.127	0.147	0.214	0.216	0.203	0.2
2	0.7	0.146	0.192	0.088	0.101	0.143	0.145	0.141	0.142
3	1.1	0.242	0.321	0.14	0.163	0.237	0.243	0.225	0.228
4	1.05	0.229	0.303	0.133	0.155	0.224	0.229	0.213	0.216
5	0.9	0.192	0.253	0.114	0.131	0.188	0.191	0.181	0.184
6	1.01	0.219	0.289	0.128	0.148	0.215	0.219	0.205	0.208
7	0.65	0.135	0.177	0.081	0.094	0.133	0.134	0.13	0.131
8	0.7	0.146	0.192	0.088	0.101	0.143	0.145	0.14	0.142
9	0.75	0.158	0.207	0.094	0.109	0.154	0.156	0.15	0.152
10	0.85	0.18	0.237	0.107	0.124	0.177	0.179	0.171	0.173
11	1.1	0.242	0.321	0.14	0.163	0.237	0.243	0.224	0.227
12	1.5	0.366	0.515	0.195	0.23	0.359	0.39	0.313	0.322
13	1.4	0.33	0.452	0.181	0.212	0.324	0.342	0.29	0.297
14	1.2	0.269	0.359	0.154	0.179	0.264	0.272	0.246	0.25
15	1.18	0.263	0.351	0.151	0.176	0.258	0.266	0.241	0.245

Figure 1 shows the allocation cost of the combinations. There is a clear indication that the individual robots are better suited for the tasks only on the basis of their allocation cost than any of their combinations. This is a problem specific condition and it largely depends on number of factors such as time and space requirement. In other words, this is mainly due to low value of workstation size and relatively high value of the speed of the robots. The load balance factor, time requirement, space requirement and allocation cost are considered for the assignment of the robots to the workstations in question.

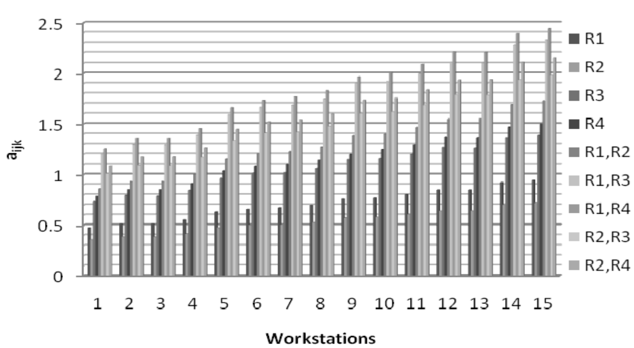

Figure 1. Allocation cost with all options

5. RESULTS AND ANALYSIS

To further examine the robustness and effectiveness of our optimization algorithm, problems are generated and tested based on our key design parameters. The optimization algorithm discussed in the previous section was coded in MATLAB for solving the linear programs. All test problems are created by a problem generator using four major design parameters: 1) the average robot service capacity (i.e., the average number of workstations that can be served by a robot based on one-dimensional resource demand of workstations); 2) the average space required by the given workstations; 3) the average machine time required by the given workstations; 4) the number of workstations to be assigned, n. For all test problems, four robot types with fixed charges are from a uniform distribution over (0, 0.41). The machine time is a weighted sum of two randomly generated values from an exponential distribution with a mean equal to 0.2. The first random variable represents the service time and the second, robot arm travel time. The results of the allocation are presented in Table 4. Overall, the computational results indicate the initial feasible solution generated by the FFOD heuristic takes no more than a second. The quality of the solution is reasonably good. The solution times for finding a near-optimum or an optimum are also recorded. As noticed, the computing efficiency is very sensitive to the problem size. However, since macro planning for a multirobot system is quite important to a designer, then the one time computing cost for optimization should not be a major concern. Thus, the algorithms developed in this paper provide significant and useful results.

Table 4. Robot selection and assigned workstation

Robot	Assigned Workstation
Robot-1	WS-1
Robot-2	WS-6, WS-14, WS-12,
Robot-3	WS-5, WS-4, WS-9, WS-10, WS-11, WS-15
Robot-4	WS-2, WS-8, WS-7, WS-3, WS-13

6. CONCLUSIONS

Multirobot facility design and planning have become increasingly important in modem production over the past decade. In this paper, a mathematical model and solution algorithm is developed to support robot selection and workstation assignment in a system employing multiple robot types. Specifically, our model considers selection of a proper mix of multiple-type robots such that operational requirements for a given number of workstations are satisfied. Each robot is characterized by its unique fixed charge and subject to its machine time and space capacity constraints. Each workstation has known time and space demands for each type of robot. The model is formulated as a pure 0-1 mathematical program, which is shown harder than the two-dimensional bin packing problem, a well-known NP-hard problem. An optimization algorithm is developed using a greedy heuristic. Computational results indicate that the algorithm is effective and efficient in solving problems of a practical size. It is worth noting that the algorithm developed in this paper has applicability to many other problems such as file placement for a storage system, using different types of storage devices, and job scheduling for a multi-processing computer system. Future research will involve both improvements in solution methods and extensions to the current model.

7. REFERENCES

[1] Dowsland K.A. and Dowsland W.B, 1992, Packing problems, European Journal of Operational Res., vol.56,pp.2-14.

[2] Johnson, D. S, 1974, Fast algorithms for bin packing, Journal of Computers and System Sciences, vol. 8, pp. 272-314.

[3] Aho, A. V, Hopcroft, J. E. and Ullman, J. D, 1974, The Design and Analysis of Comp. Algorithms. Addison-Wesley.

[4] Baker, B. S, Coffman, E. G. and Rivest, 1980, R. L, Orthogonal packings in two dimensions, SIAM Journal of Computing. vol. 9, no. 4, pp. 846855.

[5] Baker, B. S.and Schwarz, J. S, 1983, Shelf algorithms for two-dimensional packing problems, SIAM Journal of Computing, vol. 12, pp. 508-525.

[6] Brown, D. J, Baker, B. S. and Katseff, H. P, 1982, Lower bounds for online two-dimensional packing algorithms, Acta Informatica, vol. 18, pp. 207-225.

[7] Chung, F. R. K., Garey, M. R. and Johnson, D. S, 1982, On packing two-dimensional bins, SIAM Journal on Algebraic and Discrere Methods, vol. 3, pp. 66-76.

[8] Gavish, B. and Pirkul, H, 1985, Efficient algorithms for solving multi-constraint zero-one knapsack problems to optimality, Mathmatical programming, vol.31, pp.78-105.

[9] Appleton, E. and Williams, D.J, Industrial, 1987, Robot Applications, New York: John Wiley & Sons, Inc., ch. 3.

[10] Morgan, C., Robots: Planning and Implementation. England: IFS Publications Ltd, 1984.