Artificial Intelligence Design Challenge—Background, Analysis, and Relative Performance of Algorithms

Owen L. Deutsch*
Charles Stark Draper Laboratory, Inc., Cambridge, Massachusetts

The Artificial Intelligence Design Challenge was an attempt to stimulate interest in a common problem involving the application of artificial intelligence technology to problems likely to be encountered in planning, scheduling, and battle management. These problems are characterized by high combinatorial complexity, uncertainty, constraints, and, in some cases, requirements for real-time performance on finite-speed processors. Participants in the design challenge submitted competing, alternative approaches, implemented in computer code executable on desktop microcomputers, for assessment and relative evaluation over a range of problems. The range of problems was generated by variation of an input data file at the time of contest judging. The participants were given a priori knowledge of only the range of data variations, and not the specific details. In this manner, robustness to problem variations was evaluated, as was normalized performance of competing algorithms and implementations.

Introduction

THE economic availability of microprocessors with substantial throughput has enabled development of embedded applications of great ambition in the areas of planning, decision support, control of autonomous vehicles, and battle management. In some instances, these same microprocessors have enabled a revolution in engineering development environments via desktop computers. The embedded applications cited above share the common requirement of solution of mathematically hard problem formulations, usually in the sense of high complexity. Advances in algorithms, whether labeled artificial intelligence or otherwise, are as crucial to the success of this endeavor as are advances in hardware throughput and memory density. In some instances, performance improvements from clever algorithms have outstripped past and foreseeable hardware improvements by many orders of magnitude.

The Artificial Intelligence Design Challenge is focused on the design of algorithms to solve a particular hard problem, with potential applications on embedded microprocessors, and with implementation on desktop microcomputers as a medium of communication and normalized evaluation. The problem is a reformulation of the canonical traveling salesman problem (TSP), with the addition of features of statistical optimization, constraints, and relative importance of different city (goal) objectives. The problem size is limited to 11 cities, considerably smaller than the canonical problems of 10,000 cities that have been addressed in the ample literature on the TSP.[1,2] Nonetheless, with the aforementioned embellishments, the 11-city problem is far from trivial. The mathematical complexity will be described in the next section. It is noteworthy that planning and scheduling with (order) 10 multiple objectives encompass a great variety of potential real-time applications. The matrix of airfare costs between goal locations represents the results of detailed model calculations (such as the waypoint path derived from network search methods) in the (planning, scheduling, and battle management) applications.

The canonical traveling salesman problem may be stated as follows. Given a list of cities and their locations, determine the sequence of cities to be visited, (the tour) that minimizes the total distance traveled, given that each city in the list is visited at least once but no more than once. Pictorially, the problem task is to connect the dots representing each city location with the shortest length of connecting lines. Intuitively, optimal tours do not have intersecting segments and they do not double back upon themselves. Small children regularly drill in connect-the-dots-type puzzles and they are quite adept at solving TPS's by pattern recognition and intuition.

The Artificial Intelligence Design Challenge departs from the canonical TSP in the following ways. First, it is not necessary to visit all cities on the list and a city may be visited more than once. Second, the objective is to accumulate the maximum value of cities visited, where value is accumulated only on the first visit to a city. Third, the cost in traversing the distance between cities is a constraint as well as a secondary optimization objective. The cost is specified as an 11×11 matrix, with additional fixed costs allocated on each visit to a city, except for the home city. This structure rules out the use of pattern recognition, clustering, or other geometric techniques. Uncertainty is introduced by specifying that there is a probability at each transit that the salesman will have to procure a first-class passage at a specified premium. A global constraint is imposed on the minimum probability of the tour not exceeding a budget limit. Finally, a local constraint is imposed that forfeits the value accrued from the visit to a particular city unless another specified city is visited at some later position in the tour. All of these embellishments on the canonical problem are representative of the kinds of factors that arise in planning applications. They make the problem more challenging and they give impetus to the contest participants to adapt methodologies from the academic world in the direction of planning applications.

The execution and judging of the contest is specified in terms of commonly available desktop microcomputers. Such machines currently provide about 0.1 MIPS processor speed, or about 1% of the speed of large mainframe computers, along with one to several megabytes of memory, for only a few thousand dollars. In the new generation of desktop workstations, 32 bit processors running at 16 MHz clock rates, and with mathematical coprocessors, provide about 10% of the speed of mainframes, at a price between $5000–$10,000. The proliferation of these machines has enabled the networking of software and ideas between individuals by exchange of floppy disks.

In this contest, participants were required to submit a floppy disk with executable code for controlled and normal-

Received June 1, 1987; presented as Paper 87-2339 at the AIAA 1987 Guidance, Navigation and Control Conference, Monterey, CA, Aug. 17–19, 1987; revision received Feb. 1, 1988.

*Project Manager. Member AIAA.

ized evaluation by the contest judge. A set of sample data was provided in the contest specification, along with the indication that items in the input data would be changed at the time of contest judging to assess the robustness and relative performance of each submission over a range of problems. This process would verify that each submission was indeed a self-contained problem solver and was not simply repeating results that had been precalculated for a fixed problem. It discouraged method developers from tuning their implementations to perform well on one set of data at the expense of general capability.

Each submission represents complete work in the sense that an algorithm was developed and implemented, results were obtained, and a product was delivered for evaluation along with a research paper. The products are capable of solving problems that were not known a priori to the developers, except in general outline. For example, the tour budget, home city definition, first class probabilities, and airfare matrix were all subject to change at the time of judging. As will be seen in a subsequent section, changes in these numbers can radically alter the nature of the problem.

Finally, algorithm developers were free to choose the computer language in which to implement their methods. With only a requirement on the delivery of executable code on commonly available target machines, and with a time limit normalized to the most common desktop machine, the entry field included five FORTRAN-based submissions, four PASCAL-based submissions, one C-code submission, and one assembler-coded submission. Overall, this is probably a unique opportunity to evaluate alternative methodologies for solving a common problem in a controlled and normalized fashion. All of the participants are to be commended for ingenuity and perseverance. The results are not simply that the ith entry was ranked jth from the top, but that there is an interesting range of applicable methods, a fusion of ideas in the different approaches, and that great ingenuity has enabled the solution of hard problems in minutes or seconds on microcomputers when direct solution approaches require many hours on a mainframe computer for problems of this modest size. By obvious extension, larger problems in future applications of embedded computers will be solvable only with the application of similar ingenuity.

Complexity of the Problem

The canonical TSP for N cities presents a search space of possible tours of size $N!$ Since the distance matrix is unaffected by the direction of the tour, there is a twofold degeneracy resulting in only $N!/2$ unique tours. For an 11-city problem, this number is 19,958,400. The high combinatorial complexity of these problems leads to the mathematical characterization of such problems as NP-complete, implying that there are no known methods that guarantee optimal solutions in a computation time that scales as some polynomial of the number of cities. In the contest problem, the home city for the start and end of the tour is specified, and the transit costs on each leg may be different depending on the direction of travel. Furthermore, only an upper limit is specified for the total number of cities in the tour. Hence, the size of the search space is nominally

$$10! + \binom{10}{9}(9!) + \binom{10}{8}(8!) + \ldots \binom{10}{1}(1!) = 9{,}864{,}100$$

If the evaluation of each trial tour consumed only 100 ms, then direct enumeration of all possible tours would take about 550 h. When 20 city problems are considered, the projected computation times must be referenced on a geological time scale for either microcomputers or mainframe computers.

Of course, systematic pruning of the search tree is possible by the use of dynamic programming techniques. In this case, the complexity is reduced to an equally unbearable level given by

$$(N-1)(N-2)[2^{N-2}-1]$$

for the canonical problem of N cities. For the contest problem, the total number of candidate tours to be evaluated by dynamic programming is 1,129,095. Although this is an improvement over direct enumeration, the computation times and the memory required by this technique are both prohibitive for these applications. In general, the algorithmic approaches that are feasible for these problems trade computational time for optimality. The guarantee of finding an optimal solution is sacrificed for the alternate objective of finding a near-optimal solution quickly.

The statistical description of the tour costs in the contest problem adds another layer of complexity on top of the complexity of the large number of possible candidate tours. For each tour, the uncertainty in transit cost at each leg generates an event tree describing the possible costs for a single candidate tour. Hence, the evaluation of the tour cost by direct enumeration requires the expansion of 2^{M-1} branches, where M is the number of cities considered in the candidate tour. For the contest problem, this adds another loop with an iteration count of up to a 1000 inside of the loops enumerating candidate tours. This feature of the contest problem was added to represent statistical uncertainties that must be dealt with by planners in embedded applications. Planning is performed using limited knowledge of the state of the environment. For planners to be robust with respect to variability in the environment, uncertainty should be factored into the objective function. Similarly, the use of travel cost as a secondary discriminant to choose between plans of otherwise equivalent value is suggestive of practice in real planning applications. When the size of the search space leads to many plans of approximately the same evaluation, a best plan may be defined with reference to a secondary discriminant that is meaningfully related to mission objectives and constraints.

Most, if not all of the contest participants seized upon the opportunity to approximate the evaluation of the tour cost event tree. Several of the participants derived rigorous bounding criteria based on the statistics of the binomial distribution. These techniques are the most efficient for this problem, although they do not apply when the branching ratios vary between legs of the tour or if there are more than two branches for each node of the event tree. Since the costs for each leg are random variables with the same distribution, another simplification used by several participants is the application of the Central Limit theorem and use of Gaussian approximation for the distribution of tour costs. This too results in considerable computational savings, although there are approximation errors that occasionally result in the acceptance of tours that violate the budget constraint (type 1 errors) and occasional rejection of tours that actually satisfy the budget constraint (type 2 errors).

An alternative approximation for evaluation of the tour cost is a Monte Carlo approach using statistical averages of randomly sampled branches. This approach is superior in general application when the complexity of the evaluation problem is large. Frequently, sample sizes as small as 25–100 samples will yield acceptable accuracy in the evaluation of the objective function. For the contest problem, performance of three alternative techniques for evaluation are compared in Fig. 1. The computer timings were derived based on FORTRAN coding on a Macintosh Plus computer. The timings for the Gaussian approximation vary between 5.5–6.42 ms as the number of cities increase from 2–11. There is no advantage to the Monte Carlo technique for small problems, but the curves indicate that the computation time is growing exponentially for the exact evaluation and only linearly for the Monte Carlo techniques as the number of cities is increased. For the contest problem, the Gaussian approximation technique is advantageous, provided an exact evaluation is also executed at certain times to control the effects of type 1 and type 2 approximation errors previously mentioned.

Formal Problem Statement

The Artificial Intelligence Design Challenge statement was distributed at the 1986 AIAA Guidance, Navigation, and Control Conference in Williamsburg, Virginia, in August 1986. It was published several weeks later in the July–August 1986, issue of the *Journal of Guidance, Control, and Dynamics.* Copies of the problem statement were forwarded, along with letters of solicitation for contest participation, to approximately 75 individuals and groups that could be identified as working in the areas of artificial intelligence, planning and scheduling, battle management, autonomous vehicles, and advanced guidance and control. The groups contacted included most of the major academic and aerospace industry programs in applied artificial intelligence. The essence of the contest statement is reproduced in Fig. 2.

Additional clarification provided to contest participants included the notice that airfares need not be representative of any plausible fare structure, but that every fare would be nonnegative and would be less than the overall budget constraint. Moreover, the triangle inequality was not guaranteed. In other words, the fare between cities A and B was not necessarily less than or equal to the combined fares between cities A and C plus the fare between C and B. The first class premium was restricted to less than or equal to 100%. Finally, the local constraint was stated to be that the value for LAX would be accrued if and only if BOS is included at least once after LAX and not any time before LAX.

Regarding the normalization of execution time between different machines, it was stated that simple floating-point simulation benchmarks on different microcomputers had established the following scaling rules to be used during judging. The 20 min time constraint on the IBM PC would be interpreted as 5.70 min on an Apple Macintosh, 5.70 min on an IBM PC/AT (6 MHz), and 0.57 min on a Macintosh with 68020 operating at 16 MHz with math coprocessor. The equivalent time for the Commodore-64, with its 1 MHz 8-bit processor, is 68.4 min. These times are consistent with the benchmarking done between microcomputers, minicomputers, and mainframes, and across several high level languages.[3] The use of the math coprocessor on the IBM AT did not show any measurable improvement in execution time when both coprocessor and noncoprocessor versions were tested.

Description of Competing Algorithms

This section contains a capsule summary of the approach taken by each of the 11 contest participants. The order of listing follows the order of presentation in the conference session and does not denote any performance ranking. With

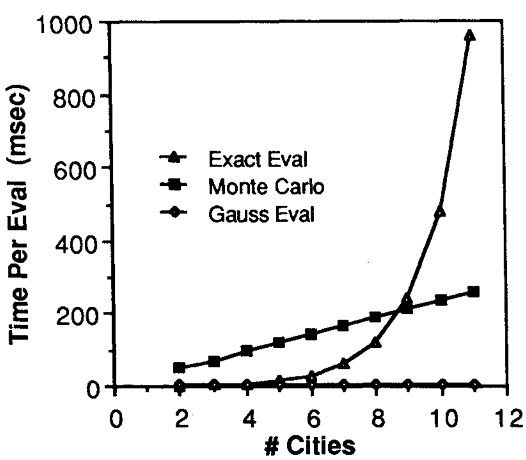

Fig. 1 Tour cost evaluation time for exact and approximate methods.

the exception of entry 3, the full papers accompany this paper later in this issue.

Entry 1 was submitted by Yuval Lirov of the Department of System Science & Mathematics at Washington University in St. Louis, Missouri. The methodology includes the use of an A^* search,[4] with branch-and-bound techniques and heuristics organized as a rule base to generate and evaluate the nodes of a search tree. The method of simulated annealing[5] is applied to compute the minimal tour of the set of cities given by the earlier steps.

Entry 2 was submitted by Karl W. Doty of The Aerospace Corporation, Los Angeles, California. The methodology uses a multialgorithm approach and a bounding procedure to evaluate the stochastic budget constraint. The tour construction methods include three different insertion algorithms: random insertion, value-order insertion, and beam search/value-cost ratio insertion. Tour modifications are performed using three more algorithms: city substitution, city position switching, and the traveling salesman solution, a specialized branch and bound procedure.

Entry 3 was submitted by Mark Peot, Stephen Cross, and Mark Fusett, all formerly of the Air Force Institute of Technology at Wright-Patterson Air Force Base, Ohio. The simulated annealing solution that was submitted for contest evaluation included a multistart approach to time-bounded computation.

Entry 4 was submitted by Prof. Manfred Padberg, on leave from the Department of Operations Research at New York University, New York, New York, and Dr. Giovanni Rinaldi of IASI-CNR, Rome, Italy. With a number of weak assumptions about the cost structure, the contest problem was formulated as a zero-one linear program, including the modeling of flow conservation and connectivity, the modeling of the budget constraint, and the modeling of the local constraint conditions. Other solution elements included a heuristic procedure, a linear program solver, a constraint or cut generator, and a branch-and-cut procedure.

Entry 5 was submitted by Clifford D. DeJong of Kaman Sciences, Colorado Springs, Colorado. The algorithm uses a set of heuristics developed for tour construction by insertion and modification. The essential insertion heuristic is city value-cost ratio. Specific attention is devoted to identifying and inserting stopover cities or hubs that reduce travel costs. Extensive use is made of integer arithmetic and quick methods to calculate the required probabilities.

Entry 6 was submitted by Roy Dahl and Karen Keating of DISTINCT Management Consultants, Columbia, Maryland; Laurence Levy and Daryl Salamone of the College of Business and Management at the University of Maryland, College Park; Barindra Nag of Towson State University, Baltimore, Maryland, and Joan Sanborn of NASA Goddard Space Flight Center, Greenbelt, Maryland. The approach combines partial enumeration to create a starting tour with a heuristic insertion procedure to expand the tour. The heuristics include lowest cost insertion and value per unit cost. Another key feature is the use of a binomial distribution to define an upper bound for the probability constraint.

Entry 7 was submitted by Prof. William H. Press and Curtis G. Callan Jr., of the Harvard College Observatory, Cambridge, Massachusetts. A segment deletion procedure is alternated with a tour regeneration step wherein tours are generated from a superposition of single city adds, subtracts, and permutes. At each step, the change is actualized by a random sampling process wherein the particular city and location involved are selected stochastically from among all possibilities, with weighting proportional to a monotonic function of the cost-value ratio. The Gauss approximation to the tour cost is made, with exact calculation on acceptance of a new best plan.

Entry 8 was submitted by John Shaw, Ron James, and Dan Grunberg of Alphatech, Inc., Burlington, Massachusetts. The methodology may be termed enumeration with extensive prun-

Artificial Intelligence Design Challenge at the 1987 Guidance, Navigation, and Control Conference

Monterey, California
August 17–19, 1987

THE challenge problem is a modification of the well-known traveling salesman problem, and is representative of a class of problems of intelligent guidance, navigation, and control systems operating in an environment of high complexity, uncertainty, and constraints.

To faciliate contest judging, all submissions included self-contained code that would execute on an IBM PC under DOS, on an Apple Macintosh, or on other desktop microcomputer systems by prior arrangement. Any programming language could be used, and only executable codes were exercised at the contest judging. Participation in the contest and the session requires submission of a floppy disk with executable code by January 30, 1987, and submission of an accompanying conference paper by February 28, 1987. The contest problem statement is:

1) There is a list of 11 cities with a value for each city and a table of intercity airline coach fares. The values for each city is credited to the salesman only if he visits that city.

2) There is an additional cost of $100 in miscellaneous expenses per city included in the tour, not including the home city at which the tour starts and ends. Nominally, the home city is Detroit.

3) There is an expense budget for the tour of $3000.

4) The objective is to construct a tour consisting of the ordered list of cities that will maximize the total valuation of those cities that are visited, while remaining within the budget constraint. For tours with the same value, the tour with the least travel expense is considered superior. It is not necessary to visit all 11 cities. A city's value is accrued only on the first visit to that city.

5) There is uncertainty in the transportation costs in that there is a 30% chance at each transit that the salesman will have to pay a 40% first-class premium to obtain a seat on the airplane.

6) The global constraint is that the planned tour must exhibit at least a 0.95 probability of not exceeding the budget limit.

7) One additional local constraint is that the valuation for visiting Los Angeles is lost unless Boston is included at some later position in the tour. This is one example of many local constraints that may appear in such problems.

It is well known that many techniques can be tuned for peak performance on a particular set of problem data. In order to compare different submissions on a common basis, and to assess robustness of performance with respect to different problem data, the submissions were required to execute the solution for contest evaluation with the following data read from and ASCI file at the time of judging, immediately prior to execution:

1) Home city definition
2) Global constraint probability and budget limit
3) First-class probability and cost multiplier
4) City valuations
5) Intercity airline coach fares.

The values given in the problem statement and tables are illustrative, with only the total number of cities remaining fixed.

Also, because performance may be a function of computation time, all submissions were constrained to an execution time equivalent to 20 min execution on an IBM PC (8088) computer without coprocessor. Solutions were judged by an independent evaluation that the constraints were satisfied, and were rank-ordered by total city value over a range of problem variations. In the event of ties, computation time and travel expense were used as secondary criteria for judging.

Intercity Airline Coach Fares

From:	ALT	BOS	CHI	DFW	DEN	DTT	LAX	MSP	MSY	PHX	SEA
To:											
ATL	–	320	220	250	330	220	600	310	150	420	550
BOS	270	–	290	410	460	230	780	310	360	580	620
CHI	190	250	–	230	260	100	640	130	240	380	450
DFW	220	490	270	–	200	330	400	250	150	250	430
DEN	390	550	300	230	–	370	290	240	300	190	340
DTT	190	200	120	280	310	–	600	170	270	440	490
LAX	500	320	270	340	250	510	–	290	430	140	270
MSP	260	370	140	290	210	200	340	–	290	340	370
MSY	170	430	280	170	350	310	520	340	–	410	630
PHX	490	690	450	300	220	520	150	410	350	–	360
SEA	660	750	530	520	280	590	320	440	530	310	–

City Valuations

City	ATL	BOS	CHI	DFW	DEN	DTT	LAX	MSP	MSY	PHX	SEA
Value	6	10	16	8	6	10	20	6	6	6	6

Fig. 2 Published contest statement.

ing by the use of bounding techniques. A simple binomial algorithm is used for calculating strict upper and lower bounds on the probability that a tour has exceeded the budget. The first step is to transform the travel costs between each city pair to the minimum costs using a shortest path algorithm. Second, bounds are calculated for the minimum number of points that can be accrued and for the maximum number of cities that can be present in a feasible tour. Tours are then enumerated from the narrowed-down field of possible tours, using the already calculated bounds and taking advantage of pruning the expansion tree wherever possible.

Entry 9 was submitted by Ting-Yi Sung, Her-Jiun Sheu, and Prof. Denis Naddef, of the Department of Statistics & Operations Research at New York University, New York, New York. The algorithm uses repeated solution of knapsack problems to determine subsets of cities to be inserted or dropped from a tour. A branch-and-bound algorithm is used, along with a variety of heuristics for tour arrangement and for addressing the specific constraints of this problem.

Entry 10 was submitted by David B. Schaechter of Lockheed Missiles & Space Company, Palo Alto, California. The algorithm is a direct enumeration of all tours, with extensive pruning of the search tree at each stage, selection of the order of enumeration so as to stumble upon good or optimal solutions earlier in the search, and very clever (assembler) programming to eliminate repeated calculations and use fast register arithmetic in the Commodore-64 microprocessor. Assembler coding achieved an apparent speed-up over the original BASIC vesion of a surprising two orders of magnitude.

Finally, entry 11 was submited by Chien Huang of Grumman Aerospace, Bethpage, New York, and Stephen Lane of the Department of Mechanical and Aerospace Engineering at Princeton University, Princeton, New Jersey. The approach uses a number of heuristics, organized as a rule-based system. The heuristics included value-cost ratio insertions, heuristics to handle local constraints, and terminal heuristics to ensure that the trip ends in the home city. Other specific heuristics attempts to identify particular situations, such as problems dominated by features of valuation, by fare structure, by the presence of a hub, and fare barriers.

Table 1 summarizes the classification of entries by method, implementation language, and target microcomputer.

Evaluation of Results

As explained in the published contest statement, the intent of the contest evaluation is to assess robustness and relative performance over a range of problem data. Robustness is defined as the ability to yield optimal or near-optimal solutions within the execution time constraints over a range of problem data. Within the allowed execution times, those entries that yield the higher value tours are considered superior. For equivalent value tours, entries with the lowest tour cost are superior. When several entries find the same tour, as in the case of the optimal tour, then the tie-breaking criterion is the execution time to yield that solution. In this evaluation, if an entry yields an optimal tour, but executes longer than entry that yields a suboptimal tour, the optimal tour result is considered superior so long as the execution time remains within the predefined time limit.

The execution time for entries run on the Apple Macintosh is compared directly with the execution time for entries run on the (6 MHz) IBM AT. Although the execution time scaling between these two machines will depend somewhat on the program and language under consideration, the very close equivalence for this problem has been established by the author. The execution time for the Commodore-64 is scaled down by the ratio of processor clock rates and bit-paths with respect to the IBM AT. Hence, all Commodore-64 timings are scaled by a factor of 1/12 to convert to equivalent IBM AT execution time for listing in tables of results.

The execution timings were obtained by stopwatch, with commencement of timing upon insertion of the problem data disk and keyboard acknowledgment. The time for loading of the entry program into RAM memory for execution was not included. For entries that displayed the best results as they were generated, these times were recorded and used in the evaluation. Whether entries self-terminated or were manually terminated on exceeding the 5.7 min time limit, the result that was recorded was the best tour until that time, and the time at which that tour was generated. A timing error of 25 s was assumed for each entry, implying that any two entries that timed to within 4 s of each other were equivalent in performance if they both yielded a tour with the same value and cost.

A total of eight problem data sets were constructed to challenge the entry programs. To condense the results over the eight problems, the following ranking system was applied. Using the criteria mentioned previously, the first-place entry for each problem was awarded 100 points, with 80, 60, and 40 points awarded to second, third, and fourth place entries. When one or more entries were tied for a position, as occurred most frequently when optimal solutions were found quickly, all tied entries were awarded the value for that position. For example, in a given problem, there might be two awards of 100 points, one for 80 points, three for 60 points, and one for 40 points.

In addition to the points awarded for relative position, there were also points awarded for absolute performance. All entries that yielded the optimal solution within 30 s execution time were awarded an additional 50 points, and all entries that yielded the optimal solution between 31 and 60 s were awarded an additional 25 points. Finding the optimal solution within these short times was considered outstanding performance, and these bonus points were an attempt to reward good performers who did not place among the top four positions in this extremely strong field of contenders. Although the ranking varied from problem to problem, in none of the eight problems did the first-place entry require more than 10 s to yield the optimal solution!

Table 1 Most methods are heuristically based

Classification of entries			
Entry	Method	Language	Computer
Lirov	A* + heuristic, annealing	PASCAL	DOS machine
Doty	Multialgorithm heuristic	FORTRAN	DOS machine
Fausett	Simulated annealing with heuristics	C	Apple Macintosh
Padberg	'0-1' linear programming	FORTRAN	DOS machine
DeJong	Cost-value heuristic	PASCAL	DOS machine
Dahl	Partial enumeration + heuristic	FORTRAN	DOS machine
Press	Heuristic, high-temp annealing	FORTRAN	DOS machine
Shaw	Bounding, enumeration	PASCAL	Apple Macintosh
Sung	Heuristic, knapsack 0-1	FORTRAN	DOS machine
Schaechter	Direct enumeration with pruning	Assembler	Commodore-64
Huang	Heuristics	PASCAL	DOS machine

The first problem, labeled Original in the tables presenting results, was the sample data provided in the contest statement. The second problem, Easy-P, was a small perturbation on the data in Original. For Easy-P, the only change was the definition of Seattle as the home city, the change of the budget limit to \$2500, and the change of the first-class probability to 50%. The Hard-P problem was also a modification of Original, but in this case the fare matrix was altered without any change in the home city, budget limit, or first-class probability. The fares to Chicago were set at \$100 from all cities, and the fares from Chicago to all cities were set to \$999. In addition to the creation of a trap at Chicago, fare barriers of \$999 were set between all cities and the home city of Detroit. The optimal tours for these three problems included nine, six, and two cities, respectively.

The problems labeled Iso-1 and Iso-2 were perturbations to Original involving both parameter and fare changes. In Iso-1, an isolated high-value situation is created. The city values were set at five for all cities except Seattle, whose value was set at 50. All fares to and from Seattle were set at \$1400, and the budget limit and probability in meeting that limit were set at \$3500 and 90%, respectively. In Iso-2, two isolated high-value cities were created. Starting from the data in Original, all city values were set at five except Phoenix and Seattle, whose values were set at 25. All fares to and from these cities were set at \$999. Finally, the budget limit was increased to \$4000. The optimal tours for these two problems included only two and three cities, respectively.

The problem labeled Ran-fare is constructed to contain a high degree of degeneracy in that many tours add up to the same value and cost. All city values are uniformly set equal to 10, and all fares are randomly sampled to be either \$200 or \$400. The budget probability constraint is 75% and the budget limit is \$2000. All other parameters are the same as in original. The number of cities in the optimal tour is six.

The problem labeled Soft-maze has a fare matrix with \$400 fares between all cities, except for one selected tour where all fares are \$100. The \$2500 budget limit is insufficient to mount this complete tour, and so the optimal tour follows this path part of the way and then tunnels through the moderate cost areas. There is also a barrier of \$999 fares between all points and the home city of Detroit. The other parameters are the same as in Original. The number of cities in the optimal tour is four.

The last problem, labeled Clusters, contains three clusters of \$100 fares between each of three cities in a matrix of \$999 fares to all other points. Cities in two of the clusters connect between clusters with \$200 fares. The remaining two cities that are not in any cluster connect to a city in one or the other cluster with a \$300 fare. The city values are all set equal to 10, except for a value of 30 for each of the two cities not in a cluster. The budget is \$4000, the first-class probability is 50%, and all other parameters are as in Original. The number of cities in the optimal tour is six.

The results are presented in the Tables 2–5. Table 2 contains the execution time in (equivalent IBM AT) seconds for each entry for each problem. If the entry terminated or was terminated without yielding the optimal solution, the entry for that position in all tables is indicated by a footnote. Entries that failed to yield any solution within the time limits are flagged by a dash. We see from Table 2 that the entry labeled Schaechter performs surprisingly well for a brute force approach. The short execution times represent a tour de force of assembler coding and insightful decomposition of arithmetic procedure. Although the approach shows no expansion potential, and would have ranked poorly if problems had been

Table 2 Execution times, s

Entry	Problem							
	Original	Easy-P	Ran-fare	Hard-P	Soft-maze	Clusters	Iso-1	Iso-2
Schaechter[a]	87	13	3	2	2	10	—[b]	2
Press	6	12	4	5	32	7	6[c]	26[c]
Doty	23	20	—	16	18	20	21	21
Lirov	80[c]	25[c]	277	3[c]	8	13	3	24
DeJong	42	4	7	—	4	—	—	145[c]
Dahl	102	24	36	3	4[c]	7	58[c]	49[c]
Shaw[d]	25	59	69	17	30	168[c]	202	165
Padberg	90	280[c]	11[c]	308	10	10	450[c]	10
Sung	40	5[c]	3	5[c]	6	3[c]	40[c]	19[c]
Fausett[d]	60[c]	60[c]	26	18	37	20	60[c]	60[c]
Huang	11	65	12	—	—	233[c]	47[c]	8[c]

[a]Timing scaled from Commodore-64 at 12:1. [b]—No solution within time limit (5.7 min). [c]Suboptimal solutions. [d]Scaled from Apple Macintosh at 1:1.

Table 3 Tour values

	Problem							
	Original	Easy-P	Ran-fare	Hard-P	Soft-maze	Clusters	Iso-1	Iso-2
Maximum value:	88	52	60	26	56	80	55	55
Maximum no. cities:	9	6	6	2	4	6	2	3
Entry								
Schaechter	88	52	60	26	56	80	—	55
Press	88	52	60	26	56	80	50[a]	45[a]
Doty	88	52	—	26	56	80	55	55
Lirov	82[a]	48[a]	60	20[a]	56	80	55	55
DeJong	88	52	60	—	56	—	—	45[a]
Dahl	88	52	60	26	52[a]	80	50[a]	45[a]
Shaw	88	52	60	26	56	80[a]	55	55
Padberg	88	44[a]	50[a]	26	56	80	15[a]	55
Sung	88	46[a]	60	24[a]	56	70[a]	45[a]	50[a]
Fausett	68[a]	46[a]	60	26	56	80	40[a]	45[a]
Huang	88	52	60	—	—	70[a]	50[a]	50[a]

[a]Suboptimal; generally, all entries get high tour values even where suboptimal.

chosen with optimal solutions containing eight or more cities, the performance is nonetheless noteworthy. It is also suggestive of an additional heuristic to exhaustively check all short tours. The entry labeled Press generally finds the optimal solution and is generally faster than most of the other entries. It did not find the optimal solution in the last two problems, however. The entry labeled Doty was the most consistent in execution time, and the most robust in terms of finding the optimal solution in all but one problem. This is consistent with the description given by Doty of his multialgorithm approach, where ultimate speed is traded for robustness. The other entries all did well, showing different relative performance in the different problems. In cases where the optimal solution was not found, near-optimal solutions were often obtained.

The tour values (the sum of the values of the cities that were visited and that satisfied local constraints) are presented in Table 3. The degree of suboptimality can be inferred for all problems, except Iso-1 and Iso-2. In these two problems, the problem constraints force the optimal tour to ignore all cities except the one or two isolated, high-value cities. The suboptimal solutions obtained by heuristic methods yielded tours where many low-value cities were visited, totally missing the obvious solution. The difference in tour value could have been exaggerated to make this point. Table 4 presents the expected tour costs for all entries for all problems. For entries yielding suboptimal solutions with the same value, the numbers in this table are a tie-breaking discriminant. Of course, if the identical suboptimal tour is found by two entries, then the tie-breaker is execution time.

Table 5, Ranking Points, is provided to summarize the performance according to the ranking system described earlier. Of course, the performance is affected by the subjective selection of evaluation problems, and by the subjective selection of a ranking system. It is fair to say that all of the entries showed great merit, but that some excelled in the subjective environment of this particular evaluation. The entries by Schaechter, Press, and Doty showed the greatest robustness and speed, although the performance of the brute force method would deteriorate substantially if problems with eight or more cities in the optimal tour were used in the evaluation. The other entries showed strength on different combinations of problems, with a grouping with respect to total score of ranking points.

The relative performance is attributable to implementation efficiencies and choice of language as well as to the underlying algorithm. It is interesting that there were many heuristic approaches that used variations on the same basic heuristics. It is also noteworthy that there is diversity in the structural framework of the methods, such as high-temperature annealing, multialgorithm, expert system with learning, linear programming, and subtour enumeration.

Summary

The Artificial Intelligence Design Challenge achieved several objectives. First, it identified a group of individuals and organizations who were capable of, and willing to undertake, the solution of a challenging optimization problem with many constraints. The particular problem is contrived, but contains features representative of real applications in planning, scheduling, and battle management.

Second, the contest showed that nontrivial optimization problems could be solved with desktop microcomputers. The stochastic optimization problem posed by this contest would

Table 4 Expected tour costs

	Problem							
	Original	Easy-P	Ran-fare	Hard-P	Soft-maze	Clusters	Iso-1	Iso-2
Budget limit:	3000	2500	2000	3000	2500	4000	3500	4000
Entry								
Schaechter	2704	2204	1844	2338	2091	3109	–	3557
Press	2704	2204	1844	2338	2091	3109	2983[a]	2827[a]
Doty	2704	2204	–	2338	2091	3109	3236	3557
Lirov	2548[a]	2200[a]	1844	2338	2091	3109	3236	3557
DeJong	2704	2204	1844	–	2091	–	–	2827[a]
Dahl	2704	2204	1844	2338	2179[a]	3109	3050[a]	3521[a]
Shaw	2704	2204	1844	2338	2091	3254[a]	3236	3557
Padberg	2704	1864[a]	1520[a]	2338	2091	3109	659[a]	3557
Sung	2704	2116[a]	1844	1643[a]	2091	2864[a]	3006[a]	3387[a]
Fausett	2594[a]	2064[a]	1844	2338	2091	3109	2996[a]	3164[a]
Huang	2704	2204	1844	–	–	3009[a]	2983[a]	3387[a]

[a]Suboptimal. Lower tour cost is tie breaker for suboptimal tours of equal tour value.

Table 5 Ranking points

	Problem								
Entry	Original	Easy-P	Ran-fare	Hard-P	Soft-maze	Clusters	Iso-1	Iso-2	Total
Schaechter		130	150	150	150	150		150	880
Press	150	130	150	150	65	150	40		835
Doty	110	110		130	110	110	130	110	810
Lirov					130	130	150	110	520
DeJong	65	150	150		150				515
Dahl		110	65	150		150	40		515
Shaw	110	65		130	65	40	60	40	510
Padberg				60	130	150		130	470
Sung	65		150	40	150				405
Fausett			110	130	25	110			375
Huang	130		130						260

[a]Press and Doty show greatest robustness and speed. All entries did well.

require many hours of mainframe computer time for solution by less clever algorithms. With the ingenuity shown by the entries in this contest, these problems were solved in only a few seconds with only 1% of the processor speed of a mainframe. The significance of this is amplified when larger problems are considered. The factorial growth of complexity leads to geological timescale computing times on even the fastest of supercomputers, unless clever algorithms are employed. Conversely, it becomes feasible to consider rapid solution of very large problems in planning, scheduling, and battle management with processors of limited throughput.

Regarding the evaluation of methodologies and implementations, it is important to emphasize again that the rankings are inherently subjective because of the selection of test problems and the ranking system that has been applied to devise a composite score. All of the entries performed well on an absolute basis, and different selections in the evaluation process would lead to different relative rankings. In the field of entries, there is an interesting convergence on heuristic methods and a surprising commonality in the use of the value-cost ratio heuristic. There are also many innovative departures, including the use of high-temperature annealing, multialgorithm structure, expert system structure with learning, bounding and subtour enumeration, and linear programming.

There is no doubt that most of the entries could be substantially improved with only minor corrections for deficiencies or omissions that have become apparent in the independent evaluation process. Also, it is likely that even greater performance will be achieved by a selective fusion of ideas from the many worthy entries. Whether all of this effort be labeled as applied artificial intelligence, operations research, or systems engineering, many embedded-processor applications become feasible in the light of these methods.

Acknowledgments

The author is indebted to several individuals for support and encouragement in the genesis and execution of this contest. Dr. Donald Fraser provided support and encouragement at all stages. Paul Zarchan provided support, enthusiasm, and technical advice. Milt Adams, James Harrison, and John McClure were also instrumental in the original concept. The author also wishes to thank Milt Adams for performing the Commodore-64 evaluation runs.

References

[1]Held, M., Hoffman, A., Johnson, E., and Wolfe, P., "Aspects of the Traveling Salesman Problem," *IBM Journal of Research and Development,* Vol. 28, July 1984, pp. 476-486.

[2]Bonomi, E. and Lutton, J.-L., "The *N*-City Travelling Salesman Problem: Statistical Mechanics and the Metropolis Algorithm," *SIAM Review,* Vol. 26, Oct. 1984, pp. 551-568.

[3]Zarchan, P., "Micros for Guidance and Control," *Proceedings of the 1987 AIAA Guidance, Navigation, and Control Conference,* AIAA, New York, 1987.

[4]Pearl, J., *Heuristics—Intelligent Search Strategies for Computer Problem Solving,* Addison Wesley, Reading, MA, May 1984, p. 64.

[5]Kirkpatrick, S., Gellatt, C.D., and Vecchi, M.P., "Optimization by Simulated Annealing," *Science,* Vol. 220, May 13, 1983, pp. 671-678.