A GENERAL BACKTRACK ALGORITHM THAT ELIMINATES MOST REDUNDANT TESTS

John Gaschnig
Dept. of Computer Science
Carnegie-Mellon University
Pittsburgh, Pa. 15213

We define a faster algorithm functionally equivalent to the classical backtrack algorithm for assignment problems, of which the Eight Queens puzzle is an elementary example (Fillmore & Williamson 1974, Knuth 1975]. Experimental measurements (figure 1) reveal reduction by a factor of 2.5 for the 8-queens puzzle (factor of 8.7 for 16 queens) in T, the number of pair-tests performed before finding a solution (i.e., first solution). A pair-test in this case determines whether a queen on square (ij, jj) attacks a queen on square (12, j2^ i^n ^{C}P^U" seconds, net speedup is by a factor of 2.0 and 6.0 for 8- and 16-queens, respectively. 16-queens was solved in 0.14 seconds on a PDP KL/10. The speedup can be attributed to the elimination of almost all redundant tests otherwise recomputed in many parts of the search tree, as indicated in figure 2, which shows the mean number of times, D, an arbitrary pair-test is executed. If D = 1 then all tests are distinct (no recomputatiOn). Note that each data point in the figures represents the mean over 30 or 70 problem instances that differ as follows: instead of instantiating queen 3, say, on square (3,1), then on (3,2),..., then (3,8), these 8 squares *are* ordered randomly. A problem instance is defined by choosing a "legal squares ordering" for each queen. Random ordering generally gives a smaller value of T, on the average, than the "natural" 1,2,3,...,N ordering (for 20-queens, a factor of 5 00 smaller!).

The algorithm exploits an advantageous time-space tradeoff and is defined below in general form by recursive SAIL procedure BKMARK [Swinehart & Sproull 1971]. The classical backtrack algorithm is defined the same, minus the underlined portions (except that "NEW[VAR]" in line 7 is replaced by "1"). The algorithm applies to any problem having NVARS variables (8, for 8-queens), each variable Xj having NVALSH] a priori possible values (8 squares for each queen (- one row of board), except 4 for queen 1 for symmetry reasons). An assignment vector ASSIGN[1:NVARS] of values to variables is a solution iff PAIRTEST0, ASSIGN[i], j, ASSIGN[j]) is true for all 0 < i < j < NVARS (iff no queen can take any other queen). Below, ASSIGN contains indices to the actual values. Top level invocation for 8-queens takes the form *tmp* <- BKMARKO, 8, A, B, C, D) with *array* dimensions D[1:NVARS] and C[1:NVARS, l:k], where k is the maximum of the B[i] values *(-8* for 8-queens). Initial values of A are irrelevant; C and D values are initially 1. BKMARK returns 1, with solution in ASSIGN, or returns 0 if no solution exists. Define PAIRTEST for 8-queens and trace the execution (new vs. old versions) to see how it works. (Suggestion: define an array VALUES with same dimensions as MARK, so that an element of VALUES encodes a board location.) For brevity, the symbol st.ands for "comment".

```
recursive integer procedure bkmark(integer var, nvars;
        integer array assign, nvals, mark, new);
begin integer i, val;       boolean testflg;
for val ← 1 step 1 until nvals[var] do
   if mark[var,val] geq new[var] then
      begin testflg ← true;
      for i ← new[var] step 1 while i < var and testflg do
         testflg ← pairtest(i, assign[i], var, val);
      mark[var,val] ← i - 1;               ! # of successful tests;
      if testflg then                      ! if passed all tests...;
         begin assign[var] ← val;
         if var = nvars then return(1)  ! done, so unwind;
         else if bkmark(var+1, nvars, assign, nvals, mark, new)
              = 1 then return(1)
         end
      end;
new[var] ← var - 1;        ! reset state of this var...;
for i ← var + 1 step 1 until nvars do  ! ...and others;
      if new[i] > new[var] then new[i] ← new[var];
return(0)                  ! backtrack and continue search;
end;
```

REFERENCES

1. Fillmore, J., and S. Williamson, "On Backtracking: A Combinatorial Description of the Algorithm", S1AM J. Computing (3), No. 1 (March 1974), pp. 41-55.
2. Knuth, D., "Estimating the Efficiency of Backtrack Programs", Mathematics of Computation (29), No. 129 (Jan. 1975), pp. 121-136.
3. Swinehart, D., and B. Sproull, SAIL, Stanford AI Project Operating Note No. 57.2, January 1971.

This research was supported by the Defense Advanced Research Projects Agency under Contract no. F44620-73-C-0074 and monitored by the Air Force Office of Scientific Research.

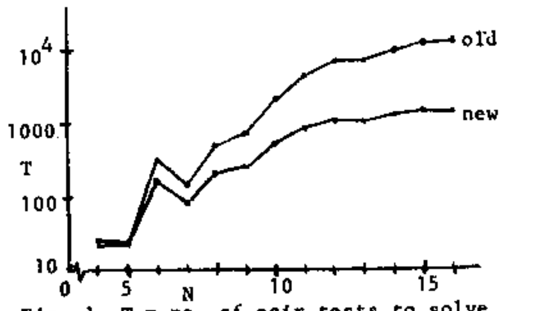

Fig. 1. T = no. of pair tests to solve N-queens puzzle
old -- classical algorithm, new -- BKMARK

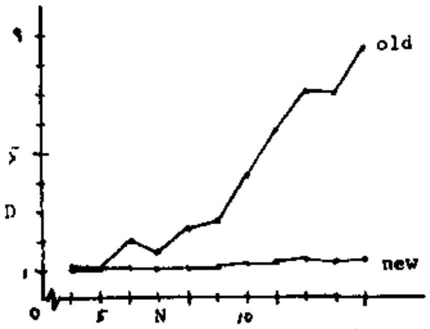

Fig. 2. D = $\frac{\text{total no. of pair tests}}{\text{no. of distinct pair tests}}$