Donald E. Knuth, Stanford University. arXiv:cs/0011047v

Let xpoints to an element of a doubly linked list. Let L[x] and R[x) point to thepredecessor and successor of that element. Then the operati ons are: L/bracketleft, R/Bracketright, L/ Bracket

L[x] and R[x) no longer have their former semantic signi*cance after xhas been removed. This fact is, of course, obvious, once it has been pointed out .

Danger lurks when objects are allowed to point int o a list from the

The element denoted by xhas been deleted from its list; why would anybody want to put it back again? Well, I admit that updates t o a data structure areusually intended to be permanent

interactive programs may need to revert to a fo rmer state when the user wants to undo an operation or a sequence of operations. Anoth er typical

Backtracking, also called depth-

The idea of (2) was introduced in 1979 by Hitotumatu and Noshi ta. It makes Dijkstra*s

Backtra ck programming can be re-garded as the task of deciding how to narrow the search and at the same time to organize the data that controls the decisions. Floyd*s elegant discussion of the connection between backt racking and nondeterminis-tic algorithms includes a precise method for updating d ata structures before choosing

In simple situations we can simply maintain a stack that cont ains snapshots of the relevant state information at all ancestors of the current n ode in the search tree. But the task of copying the entire state at each level might take too m uch

1

Dijkstra*s recursive procedure for the queens problem kept the current state in three global Boolean arrays. Hitotumatu and Noshita* s program kept it in a doubly

When Dijkstra tentatively placed a queen, he changed one ent ry of each Boolean array from true to false; then he made the entry true again when back tracking. Hitotumatu and Noshita used (1) to remove a column and

The beauty of (2) is

General schemes for undoing assignments require us to recor d the identity of the left-handside together with its previous value. But in this case only the single quantity xis needed

We can apply (1) and (2) repeatedly in complex data structure s that involve large numbers of interacting doubly linked lists. The program log ic that traverses those lists and decides what elements should be deleted can often be run i n reverse

This process causes the pointer variables inside the global data structure to execute anexquisitely choreographed

Given a matrix of 0s and 1s, does it have a set of rows containing exactly one 1 in each c olumn? We can think of the columns as e lements of a universe, and the rows as subsets of the universe; then the problem is to cover the universe.

Dana Scott conducted one of the *rst experiments on backtrac k programming in 1958. His program, written for the MANI

12 pentominoes are placed in

Scott was probably inspired by Golomb*s paper [14] and some extens ions reported by Martin Gardner. For example, one of the 65 solutions is shown in Figure

construct all possible rows representing a way to place a pentomino on the board. Each row contains a 1 in the column id entifying the piece, and 1s in the columns identifying its positions.

Algorithm X is simply a statement of theobvious trial-and-error approach. Algorithm X solutes all solut ions to the exact cover problem.

IfAis empty

Choose a row,

Include rin the partial solution

For each j such that A[r, j] =

Repeat this algorithm rec

The nondeterministic choice of rmeans that the algorithm essentially clones itself into subalgorithms. Each subalgorithm inherits th e current matrix A, but reduces

The subalgorithms form a search tree in a natural way, with the original problem

Backtracking is the process of traversing the tree in preord er, *depth *rst.* Any systematic rule for choosing column cin this procedure will provide all solutions. But some rules work much better than others.

Scott realized that piece X has only 3 essentially di*erent positions, namely centered at 23, 24, and 33. The full set of 8 ×65 = 520 solutions is easily obtained by rotation andre*ection.

Golomb and Baumert suggested choosing, at each stage of a backtrack procedure, a subproblem that leads to the fewest branches. In the case of an exact cover problem, this

Search trees for Scott*s pentominoproblem then have only 10,421 nodes (X at 23);

Rows of the matrixare doubly linked as circular lists via the LandR*elds. Each columnlist also includes a special data object called its list header.

The list headers are part of a larger object called a column object. The size is the number of 1s in the column, and the name is a symbolic identi*er for printing the answers.

TheLandR*elds of the list headers link together all columns that stil l need to becovered. This circular list also includes a

The 0-1 matrix of (3) would be represented by the objects shown in Figure 2. For example, if we name the columns A, B, C, D, E, F, and G, this is the matrix. The Clinks are not shown because they would clutter up the picture.

Cover column c(see below)

For each r, while r/negationslash=c, setOk*r; for each j, while j/negationlash=r, uncover column j(see

Uncover column c(see below) and return

The operation of printing the current solution is easy. We su ccessively print the rows containing data object O0,O1,. . .,Ok*1.

5

Four-way-linked representation of the exact

To choose a column object c, we could simply set c*R[h]; this is the leftmostuncovered column. Or if we want to

The operation of covering column cis more interesting: It removes cfrom the headerlist and removes all rows in c*s own list from the other column lists they

SetL/bracketleftbig

For each i, while i/negationslash=c, set S/bracketleftbigC[j]/br bracketrightbig. For each j, set R/brackleftbig

Operation (1) is used to remove objects in both the horizontal and

For each i=U[c],U/bracketleftbig U[c]/brackrightbig, while i/negationslash=c, we get to the point of this whole algorithm, the oper ation of uncovering a givenColumn c. Here is where

R/bracketleftbig

6

Uncovering takes place in precisely the reverse order of the covering operation. Notice that (2) undoes (1)

Figure 2 shows what happens when search (0) is applied to the data of (3) Figure 3 shows the asymmetry of the links that now appear in column D.

Search (1) will cover column B, and there will be no 1s left in column E. Continuing search (0), when rpoints to the A element of row (A ,D,G), we also covercolumns D and G. So search (2) will *nd nothing.

7

The links after columns D and G in Figure 3 have

If the S*elds are ignored in the choice of c, or if the shortest column is chosen at each step, the solution will be found.

Readers who p lay through the action of this algorithm on some examples

The running time of algorithm DL X is essentially proportional to the number of times it applies operation (1) to remove an ob ject from a list. E*ciency considerations. When algorithm X is implemented in terms of dancing links, let*s call it algorithm DLX.

The search tree for one

Study larger examples before drawing any general conclus

A backtrack program usually spends most of its time on only a f ew levels of the searchtree. Figure 5 shows the search tree f or the case X = 23 of Dana Scott*s pentomino problem using the Sheuristic.

More than half of the nodes lie on levels *8. Extra work on the lower levels has reduce d the need for hard work at the higher levels.

Each update involves about 14 memory accesses when the Sheuristic is used, and about about 8 when Sis ignored.

The heuristic is even more e*ective in larger probl ems, because it tends to reduce the total number of nodes by a factor that is exponenti al in the number of levels. The cost of applying it grows only linearly.

The Sheuristic is good in large trees but not so good in small ones. I tried a hybrid scheme that uses the Sheuristic at low levels but not at high levels. This experiment was, however, unsuccessful.Therefore I decided to retain the sheuristic at all levels of algorithm DLX.

10

My trusty old SPARC station 2 computer, vintage 1992, is able to perform approxi - mately 0.39 mega-updates per second when working on large pr oblems. The 120 MHz Pentium I computer that Stanford computer science faculty were given in 1996 did 1.21 mega- updates per second.

Dancing links can be used to solve complex problems with simple geometric structure. The technique of dancing links is actually a step backw ard from Scott*s 40-year-old method.

The task of packing the set of pentominoes into a 6×10 rectangle is more difficult than Scott*s 8 ×8*2×2 problem. The backtrack tree for the 6 ×10problem is larger and there are 2339 essentially di*erent so lutions [21]. In this case we limit the Xpentomino to the upper left

John G. Fletcher needed only ten minutes to solve the 6 ×10 problem on an IBM

The 7094 had a clock rate of 0.7 MHz, and it could access two 36- bit words in a single clockcycle. Fletcher*s program required only about 600 ×700,000/28,320,810 clock cycles.

Many people have written about polyomino problems, includi ng distinguished math-ematic

92 solutions, 14,352,556 nodes, 1,764,631,796 updates. 100 solutions, 10,258,180

Algorithm DLX will branch on the ways to place a cell if some piece is di*cult to place. It knows no di *erence, because pieces and cells are simply columns of the given inpu t matrix.

Algorithm DLX begins to outperform other pentomino-placin g procedures in problems where the search tree has many levels. For example, let*s con sider the problem

The Multum produced an answer after more than an hour, but she remained uncertain whether other solutions were possible. Now, with the dancin g links approach describedabove, we can obtain several solutions almost instantly. The solutions fall into four classes,

12

In the late 1950s, T. H. O*Beirne introduced a pleasantvariation on polyominoes by substituting triangles for squ ares.

The twelve hexiamonds were independently dis covered by J. E. Ree

The 6 ×6 rhombus can be tiled by the twelve hexiamonds in exactly 156 ways. Figure 7 shows one such arrangement, together with some arro w dissections.

13

O*Beirne was particularly fascinated by the fact that seven of the twelve hexiamonds have di*erent shapes when they areipped over. In November of 1959, after three months of tri als, he found a solution; and

A 19-level backtrack tree with many possibilities at each le vel makes an excellent test case for the dancing links approach to covering the puzzle. I broke the general case into seven subcases, depen ding on the distance of thehexagon piece from the center. The total number of updates performed was 134,425,768,494.

My goal was not only to count the solutions, but also to count arr angements that were as symmetrical as possible. The overa ll hexagon has 156 internal edges, and the 19 one-sided hexiamonds have 96 internal non-edges.

A solution to the hexiamond problem is maximally symmetric if it has the highest horizontal or vertical symmetry score. Four of the eight solutions shown in Figure 8 have a horizontal symmetry score of 3 2. John Conway found these solutions by hand in 196 4 and conjectured that they were symmetric overall.

14

Figure 8 shows the solutions to O*Beirne*s hexiamond hexagon problem. The small hexagon is at various distances from the center of the large one.

15

There are 46 ways to pack the one-sided pentominoes in a 3 ×30rectangle. Figure 9 shows a maximally symmetric ex ample (which isn't really very symmetrical)

46 solutions, 605,440 nodes, 190,311,7

I set out to count the solutions to the 9 ×10 problem. I soon found that the task would be hopeless, unless I invented a much better algorithm. The Monte Carlo estimation procedure of [24] sug gests that about 19 quadrillion updates will be needed, with 64 trillion nodes in the search

I do, however, have a conjecture about the

A failed experiment. Special arguments based on *coloring* often give important in- sights into tiling problems.

For example, it is well known that if we remove two cells from opposite corners of a chessboard, ther e is no way to cover the remain- ing 62 cells with dominoes.

Algorithm DLX makes 4, 780,846 updates before concluding that there is no

The cells of the hexiamond-hexagon problem can be colored bl ack and white in a similar fashion. All triangles that point left are black, sa y, and all that point right are white. The remaining four, namely the *sphinx* and the *

I expected the subproblems to run up to 16 times as fast as the o riginal problem. I expected the extra information about impossible correlati ons of piece placement to help algorithm DLX make intelligent choices.

The overallproblem had 6675 solutions and required 8,976,245,858 upda tes. The sixsubproblems turned out to have respectively 955, 1208, 1164, 1106, 1272, and 970 solutions.

Brian Barwell considered making tetrasticks from lin e segments or sticks. He called the resulting objects polysticks. There are 2 disticks, 5 tristicks, and 16 tetrastick.

Barwell proved that the sixteen tetrasticks cannot be assem bled into any symmetricalshape. But by leaving out any one of the tetrastick that h ave an excess of horizontal or vertical line segments, he found ways to create a 5 ×5 square. Such puzzles are quite di*cult to do by hand, and he had found only *ve solut ions at the time he wrote his paper.

Generalized problem asks for a set of rows that covers every primary column exactl y once and every secondarycolumn at most once.

The tetrastick problem of Figure 11(c) can be set up as a gener

Figure 11 . Filling a 5×5 grid with 15 of the 16 tetrasticks; we must leave out either the H, the J, the L, the N, or the Y.

W, X, Y, Z representing the *fteen tetrasticks (excluding L) Columns H xyrepresenting the horizontal segments (x, y)**(x+1, y), and V xy representing the vertical segments. Secondary columns I xytorepresent interior junction points (x,. y), for 0 <

Polysticks are not supposed to

The common interior point I33 means that these rows cross eac h other. For example, the two rows corresponding to the

Figure 11(c) covers only the interior points I14, I21, I

We can solve the generalized cover problem by u sing almost the same algorithm as before. The only di*erence is that we initializ e the data structure by making a circular list of the column headers for the primary columns only. The header

A generalized cover problem can be converted to an equivalen t exact cover problem. But we are better o* working with the generalized proble m, because the

There are ten one-sided welded tetrasticks if we add the mirror images of the unsymme trical pieces. Only three solutions are possible, i ncluding the two perfectly symmetric solutions shown.

One-sided welded tetrasticks

There are 14 one-sided unwelded tetrasticks, and I thought they would surely fit into a 5×5 grid in a similar way. But this turned out to be impossible. T he reason is that four of the six pieces J , J*, L, L*

Figure 14 shows

19

I also tried unsuccessfully to pack all 25 of the one-sided te trasticks into the Aztec diamond pattern of Figure 15. I see no way to prove that a

The 4 queens problem is just t he task of covering eightprimary columns (R0 ,R1,R2,R3,F0,F1,F2,F3) corresponding to ranks and *les. The problem is actually a special case of the generali zed cover problem in the previous section.

When we apply algorithm DLX to this generalized cover proble m, it behaves quite differently from the traditional algorithms for the Nqueens problem. It branchessometimes on di*erent ways to occupy a rank of the chessboard and sometimes on di *les of a chessboard.

Central positions rule out more po ssibilities for later placements. We gain e*ciency by paying attention to the order in which primary columns are considered.

Figure 16(a) shows an empty board with 8 possible ways to occupy each rank and each *le. Placing a queen in R2 and F3 after Figure 16(d) m akes it impossible to cover F2. Backtracking will occur even though only four qu eens have been tentativelyplaced.

Figure 16. Solving the 8 queens problem by treating ranks and *les symm etrically.

21

The order in which header nodes are linked together at the sta rt of algorithm DLX can have a signi*cant e*ect on the running time. For example, the search tree has 312,512,659 nodes and r equires 5,801,583,789 up-dates, if

Algorithm DLX solved small cases of the Nqueens problem using organ-pipe order. The advantage of mixing rows with co lumns becomes evident as the number of solutions increases.

Special methods are known for countin g the number of solutions to the Nqueens

Algorithm DLX is an e*ective way to enumerate all solutions to such problems. On small cases it is nearly as fast as algori thms that have been tuned to solve particular classes of problems.

The orde ring heuristic is used to tackle larger and la rger cases all the time

In this paper I have used the exact cover problem to illustrat e the versatility of dancing vehementlylinks. I could have chosen many other backtrack applicat ions in which the same ideas apply. For example, the approach works nicely with the Waltz *ltering algorithm [36]

I wish to thank Sol Golomb, Richard Guy, and Gene Freuder for t he help they generously gave me as I was preparing this paper. Ma ggie McLoughlin

"I profoundly thank Tomas Rokicki, who provided the new co mputer on

My names for the tetrasticks are slightly di*erent from those originally proposed by Barwell. I prefer to use the letters J ,R,and U for the pieces he called U, J, and Crespectively.

The implementation of algorithm DLX that I used when prepari ng this paper is *le dance.w on webpage http://www-

23

This puzzle, whichis available from www.puzzletts.com , actually has only 83 solutions. It carries a Chinese title, *Dr. Dragon*s Intelligence Pro*t System.*

[1] Brian R. Barwell

Elwyn R. Berlekamp, John H. Conway,

Max Black, Critical Thinking (Eng

Ole-Johan Dahl, Edsger W. Dijkstra, and C. R. Ho

N. G. de Bruijn, personal communication (9 September 199 9): *. . .it was almost my *rst activity in programming that I got all 2

*I could speed the matter up by having a very long program, a nd that one wasgenerated by

Henry Ernest Dudeney, *74*The broken chess

A program to solve the pentomino prob lem by the recursive

Robert W. Floyd, *N

Martin Gardner, *Mathematical games: More about compl ex dominoes

Michael R. Garey and David S

Solomon W. Golomb, *Check

Solomon W. Golomb, Polyomino

Solomon W. Golomb and Leonard D.

24

Richard K. Guy, *Some mathematical recreations,*

Richard K. Guy, *O*Beirne*s Hexiamond,

Robert M. Haralick and Gordon L. Elliott, *Increasing t

Packing a square with Y-pentomin oes

C. B. and Jenifer Haselgrove, *A

Hirosi Hitotumatu and Kohei Noshita, *A

George P. Jelliss, *

Donald E. Knuth, *Estimating the

Donald E. Knuth, T

Jean Meeus, *Some polyom

N. Metropolis and J. Worlton, *

Pell*s equat ion in two popular problems

T. H. O*Beirne, *Puzzles and Paradoxes 44: Pentominoes and hexiamonds,* NewScientist 12(1961), 316*317.

Hexiamond is a type of diamond

J. E. Reeve and J

Igor Rivin, Ilan Vardi, and Paul

Dana S. Scott,*Programming a combinatorial puzzle,* T echnical Report No.1 (Prince-ton, New Jersey: Princeton

University Department of

P. J. Torbijn

David Waltz, *Understanding line drawings of scenes wi th shadows,* in

Bernhard Wiezorke and Jacques Ha

Alfred Wassermann of Universit¨ at B ayreuth covered the Aztec diamond of Figure 15 with one-si ded tetrasticks. The

107 poss ible solutions, which

Many of these turn out to be more symmetric than the o ne in Figure 10