DAVID CLARK & MALCOLM BROOKS

THE SWISS EVENT
at the University of Canberra Maths Day

An event based on 'find the rule' problems has been part of the University of Canberra Maths Day since 1985. This paper discusses the type of problems used, and how they are used in an event at the maths day. It also discusses how the event evens out the competition and stops strong schools dominating.

Introduction

Try this problem! From the table below, deduce the values of y for the extra values of x; or, equivalently, deduce the function $y = f(x)$.

x	8	4	20	17	40	7	1	25	0	…
y	19	11	43	37						

'Find the rule' problems such as this are used in the 'Swiss' event, one of the events that comprise the University of Canberra Maths Day. The Swiss event was introduced into the maths day in 1985. In this paper we will describe the Swiss event, its context, its aims, its logistics and give examples of problems used in past events.

The University of Canberra Maths Day

Since 1982 the University of Canberra has run a maths day for final year secondary students from schools in the Canberra region. It is held in the University of Canberra gymnasium and is sponsored jointly by the University, its School of Mathematics and Statistics, and the Australian Mathematics Trust.

The UC Maths Day aims to be an exciting, enjoyable, light-hearted but nonetheless challenging day which provides an opportunity for mathematically able students to celebrate their talents. Up to forty teams, each of five students, compete in a number of events and there are trophies and prizes for the winning team overall, and for the teams with the greatest improvement over their school's performance in previous years.

Each team is accompanied by a teacher (who supervises the team from another school). The teachers are encouraged to take back ideas and materials from the day's events to use in the classroom.

There are five events.

- ***Poster Contest***
  An open-ended challenge problem is sent to teams a month in advance and teams present their solutions on large, colourful and often amusing posters which are displayed and judged on the day.
- ***Group Contest***
  Teams work together on a set of ten story-type problems.
- ***Swiss Contest***
  Pairs of teams compete in 'find the rule' problems as described in this paper.
- ***Cross Contest***
  Students attempt to solve a crossnumber puzzle. Each team is split into two, with one half seeing the across clues and the other half seeing the down clues.
- ***Relay Contest***
  Each team is split into two and the halves alternate in attempting the next available question from a set of twenty. Questions are on a range of topics and at a variety of levels of difficulty. Questions can be attempted several times and/or abandoned but must be done in sequence.

More details can be found in the Maths Day books [1], and in the paper describing the Cross Contest [2].

There is a wide range of ability in the students representing their schools. Some of the larger city schools have students up to Mathematics Olympiad ability, while the smaller country schools may have to send the whole of their Year 12 advanced mathematics class. The UC Maths Day adapts to this disparity in a number of ways. For the Maths Day as a whole trophies are awarded for the most improved city and country schools as well as for the overall winner. Further, each event has specific ways of adapting to the disparity in students' abilities. Those of the Swiss event are described in this paper.

'Find the rule' problems

In find the rule problems, students have to relate a y value with a given x value. Or, equivalently, to find a function f so that $y = f(x)$. Students are given an initial set of four (x, y) pairs, and are asked to find the y values corresponding to an additional sixteen x values. A simple example was given in the introduction (where, of course, $f(x) = 2x + 3$).

It is something of a balancing act to construct problems of the appropriate level of difficulty. If the problem is too easy, there is no challenge. If it is too hard, students may become frustrated or discouraged. While the above example may be too simple, particularly

Problems

1996, number 2																
x	100	30	600	350	540	70	710	111	321	777	666	555	888	620	339	594
y	14	4	85	50												
2000, number 4																
x	3	4	15	7	9	6	12	25	1999	30	49	11	999	70	125	99
y	2	0	5	5												
1997, number 5																
x	9	15	36	63	18	81	50	55	80	90	100	77	99	111	222	333
y	10	16	40	70												
1997, number 2																
x	4	5	11	12	6	3	9	8	10	7	13	15	14	25	19	17
y	61	52	121	441												
1992, number 2																
x	1	2	3	5	4	21	53	74	84	139	149	181	-73	226	228	1000
y	2	3	5													

after the results of a few more guesses are known, a problem which seems obvious to its constructor can seem far from obvious to others. We address this by careful moderation and by providing hints. Each problem is provided with a cryptic hint and sometimes a broad hint. How and when these hints are provided has varied over the years. Our current practice is to print the cryptic hint on the display sheet and to give the supervisor discretion on when and if to give the broad hints.

Below are some examples of problems which have been used in the past, together with their cryptic hints and solutions. So that the reader can get a feel for solving the problem, all of the problems are given, then the next few y values, then the cryptic hints, then the solution.

Next four y values

1996, #2:	77, 10, 101, 15
2000, #4:	2, 2, 8, 0
1997, #5:	20, 100, 55, 61
1997, #2:	63, 9, 18, 46
1992, #2:	5, 23, 59, 79

Cryptic hints

1996, #2:	A dwarf's share?
2000, #4:	A Y2K problem?
1997, #5:	A game for the under-10s, basically.
1997, #2:	It all depends on which way you look at it.
1992, #2:	Farmers (and other primary producers) should do well in this round.

Solutions

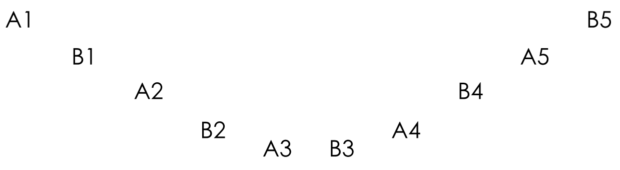
1996, #2:	$y = x$ div 7
2000, #4:	$y = 2000 \bmod x$
1997, #5:	$y = x$ written to base 9
1997, #2:	$y = x^2$ backwards
1992, #2:	y is the first prime greater than x

The logistics of the Swiss event

The Swiss event at the UC Maths Day is organised as a series of 4 to 6 rounds depending on the time available. In each round a school competes against one other school. Thus each round consists of 15–20 separate 'matches'. This is in contrast to the other events where each school competes directly against every other school.

In each match the students from the two schools are interleaved. They sit in a crescent with the *(x, y)* values printed on a large sheet of paper on the floor in front of them.

x	8	4	20	17	40	7	1	25	0	…
y	19	11	43	37						



Students take turns in guessing the next y value. If the guess is correct a point is awarded to the student's school. There is no penalty for an incorrect guess. All students hear each guess and the supervisors 'right' or 'wrong' response. Irrespective of whether a student's guess is right or wrong, the student has to wait until all other nine students have had a turn before guessing again. Students continue to guess the y value corresponding to a particular x value until a guess is correct. At this point the new y value is written onto the sheet. Then they move onto the next x value. If students are stuck, the supervisor may give hints (see 'Rationale for the Swiss event' below).

The pairings at the Swiss event

Round robin competitions are familiar to followers of games ranging from football to chess. They are organised as a series of rounds, each of which consists of a number of matches. In a single round robin competition contestants play each other exactly once and the number of rounds is one fewer than the number of contestants.

'Swiss' competitions were introduced by the Swiss chess federation to enable a large field of

Photo courtesy University of Canberra

candidates to be ranked without having a full round robin. There is a great deal of flexibility in choosing who will play who in the first round. In choosing the pairings for subsequent round two considerations need to be taken into account. Firstly no contestant should play the same opponent more than once. Secondly, the contestants are ranked more effectively if the pairings are of contestants with roughly equal ability.

At the UC Maths Day there is already a ranking based on the schools' performances in the Group event. In the first round the top team is paired with the second team, the third with the fourth, etc. In subsequent rounds the pairings minimise the difference in rankings, subject to no school competing against the same school more than once. The rankings are updated between rounds. More details are given in 'The Swiss Algorithm' below.

Victory points

Victory points are widely used in bridge competitions. The aim is to prevent a blow-out in scores if one team happens to dominate another. When victory points are used the total number of victory points is fixed. If there were, say, 20 victory points available and one team was awarded 14, then their opponents would be awarded 6. Awarding victory points is a two stage process. Firstly raw points are decided by the results of the match. Then a formula is used to convert the raw points into victory points. The raw points relate to the context of the competition. For instance, when Swiss scoring is used in contract bridge, the raw points are the difference in points earned by the two teams in a hand of bridge. In our 'find the rule' problems, the raw points are the number of correct answers by each team. These are converted to victory points using the formula 6 ± the difference in raw scores, with a maximum of 12 and a minimum of 0.

Rationale for the Swiss event

There is a wide range of ability in the students who compete at the UC Maths Day. This difference in ability is both between schools and within schools. One of the aims of the organisers is to give a sense of achievement to all students from all schools. In the Swiss contest, this is addressed in a number of ways.

Firstly, advantage is taken of the number of rounds being much smaller than the number of teams. The algorithm described below ensures that the difference in ranking of competing schools is minimised. This makes it unlikely that there will be a mismatch between schools.

Secondly the use of victory points rather than raw scores puts a limit on the spread of points awarded for the event. In particular, it prevents a blow-out if there is a mismatch in one of the matches. It also prevents a good school for benefiting too much if they happen to be paired with a weak opponent.

Thirdly, because students are interleaved, the next guess is always by a member of the opposing team. Further, having students wait before answering their next question also has the benefit of preventing a quick thinking student from dominating others in the mini-competition. This includes students from his or her own school as well as the opposing school.

Finally, supervisors have discretion about when to give the cryptic hint and the broad hint. Hints may be given earlier if the students seem to be struggling. In addition, supervisors may make up their own hints if the ones supplied prove ineffective. This does not disadvantage any other team as the hints are only heard by students in teams who are directly competing against each other, and the scoring system uses the difference in the number of questions answered correctly.

The Swiss Algorithm

In order to run the Swiss event it is necessary to find the pairings of schools for each round. Finding pairings which do not conflict with previous rounds' pairings is not difficult, and solutions are well known. They have been used in contract bridge competitions for several decades. What makes finding the pairings more difficult for the UC Maths Day is the requirement of not pairing teams with widely differing abilities. A further problem is that the rankings are updated between rounds, so that it is not possible to find pairings for all rounds based on the teams' rankings in the previous event, the Group Contest (the Swiss Contest is the second event of the day). On the other hand, there are only 4 to 6 rounds, so that the problem is still manageable.

When the Swiss event was first introduced, one of the organisers of the UC Maths Day would work out the pairings by hand. Having run many bridge competitions he is highly skilled at finding pairings. Nevertheless, it becomes non-trivial as the number of rounds increases, and given the necessity of entering marks for the previous round and the overhead of relocating teams between rounds we often got through fewer rounds of the Swiss than we had prepared.

Recently one of the authors wrote a small computer program to do the pairings between rounds. This was incorporated into the score-keeping software used on the maths day. This has worked well since, as it eliminates the time taken to determine the pairings. The delay caused by entering the results has also been eliminated by determining the pairings on the rankings one round further back. That is, the pairings for the first two rounds of the Swiss are based on the team rankings after the Group contest, the pairings for the third round of the Swiss is based on the team rankings after the first round of the Swiss, etc. This allows the scores for a round to be entered as the students are doing the next one.

A branch and bound algorithm was developed to find the pairings. It is described in the appendix.

Where to get more

The University of Canberra Maths Day home page can be found at
http://www.blis.canberra.edu.au/
MathStat/frames/maths_frset.htm.

References

Brooks, M. S. (Ed.). University of Canberra Maths Day books for the years 1985 to 1991 are available from the Australian Mathematics Trust, PO box 1, Belconnen ACT 2616.

Clark, David (2002). Crossnumber Puzzles at the University of Canberra Maths Day. *The Australian Mathematics Teacher 58* (3), 24–29.

The University of Canberra Maths Day home page http://www.canberra.edu.au/mathsday

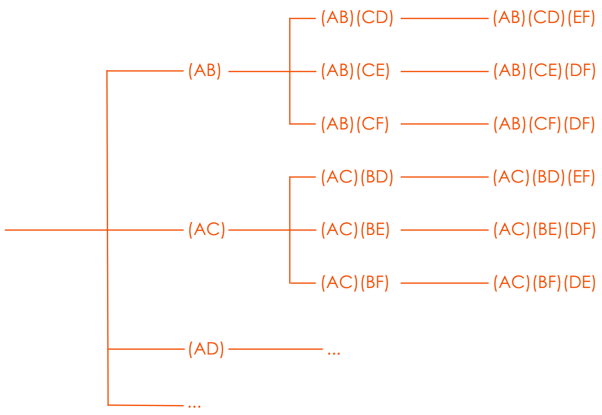
Appendix: The branch and bound algorithm to program to determine Swiss pairings

The cost function

The Swiss algorithm finds the pairings with the lowest cost. Suppose the teams are t_1, t_2, …, t_n (in rank order). Then the cost of a match between t_i and t_j is $10^{|i-j|} - 1$ if t_i and t_j have not already competed against each other, and ∞ if they have. The cost of the whole pairing is the sum of the costs of the individual matches. For example, a paring of $\{(t_1, t_4), (t_2, t_3), (t_5, t_7), (t_6, t_8)\}$ would have a cost of 121. A useful consequence of this cost formula is the information it gives. For example, with 34 teams a cost of 359 means that the pairing has 3 matches where the contestants are 3 apart in ranking, 5 matches where the contestants are 2 apart in ranking and 9 matches where the contestants are 1 apart in ranking.

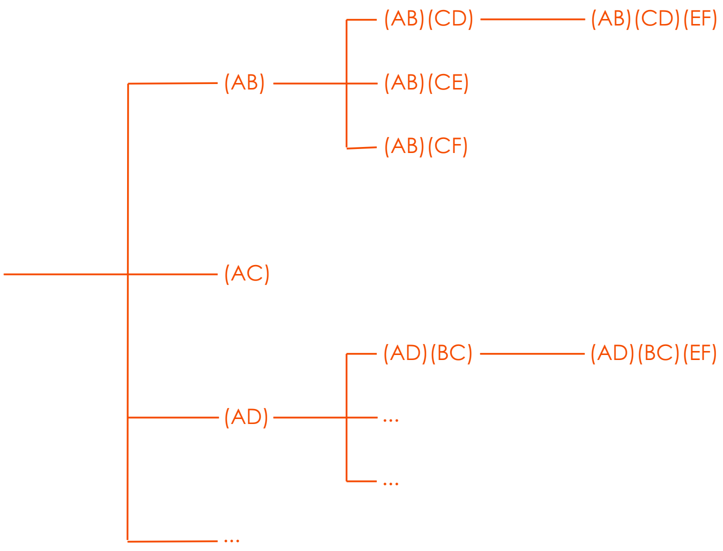
The branch and bound algorithm

The algorithm used is a classic branch and bound algorithm. Pairings are built up in a systematic way. All possible pairings are evaluated using the cost function, and the best is chosen. This is the 'branch' part of the algorithm. The diagram below shows part of the tree of solutions for a 6 teams (3 pairs) competition, with teams A, B, C, D, E and F.



Constructing and evaluating all possible pairings is not practicable. Even for only 30 schools there are about 6×10^{15} possible pairings. At one million per second this would take 200 years. We are not convinced that we could keep the students' attention for quite that long between rounds.

Instead, when the cost of a partial pairings exceeds the cost of the best solution so far, it is abandoned. This is the 'bound' part of the algorithm. The diagram below demonstrates this. An initial solution {(AB)(CD)(EF)} is found. The cost of the partial solutions {(AB)(CE)} and {(AB)(CF)} both exceed the cost of {(AB)(CD)(EF)}, so their solutions are not completed. Then the branch {(AC)} is entered. It is found that A and C have played against each other in a previous round, so that whole branch is abandoned. The next full solution evaluated is {(AD)(BC)(EF)}.



The algorithm was implemented in JavaScript. The source code is available from the first author.

David Clark
School of Information Sciences and Engineering
University of Canberra
davidc@ise.canberra.edu.au

Malcolm Brooks
School of Information Sciences and Engineering
University of Canberra
malcolmb@ise.canberra.edu.au