Olympiads in Informatics, 2009, Vol. 3, 38–59
© 2009 *Institute of Mathematics and Informatics, Vilnius*

Infrastructure for Contest Task Development

Rob KOLSTAD
USA Computing Olympiad
15235 Roller Coaster Road, Colorado Springs, CO 80921, USA
e-mail: kolstad@usaco.org

Abstract. The USA Computing Olympiad annually conducts six internet-based computer programming competitions, each including three to four algorithmic tasks in each of three divisions. Coupled with the training camp competitions, a typical annual USACO 'problem budget' approaches 75 new tasks at three distinct levels of difficulty.

In order to exploit a distributed coaching staff, USACO developers created and evolved the web-based 'probgate' problem-development system to speed production of acceptable quality programming contest tasks that are machine-gradable, well-accepted, and yield no or few complaints, regrades, or requests for clarification.

This paper describes each of the major modules and shows how they are used to simplify, speed up, and automate administration of contests regularly accessed by more than 1,000 students.

Key words: programming contests, automation, contest test data, contest data validation, contest preparation, contest automation, automatic grading.

1. Demographic Background

The USA Computing Olympiad is the USA's designated organization for training and selecting students to compete in the International Olympiad in Informatics.

In July, 2009, the United States had a population of just over 307 million people. The USA includes 9.8 million square kilometers (about half the size of Russia or South America; about twice the area of the European Union). The 3,500 mile (5,600 km) width of the contiguous 48 states complicates gathering students for training camps or on-site contests.

The USA has just over 29,500 high schools with a total of about 15M students; about 3.2M will graduate in 2009. Additionally, 0.3M high school students are 'home schooled' outside the traditional public and private school system.

Until recently, USA high schools included all levels of students and declined to utilize 'tracking' to segregate them by performance. The past decade has seen increased availability of 'magnet,' 'gifted and talented,' and 'charter' schools that offer focused or deeper programs for academically inclined students. Additionally, many high schools offer 'Advanced Placement,' 'Honors,' or International Baccalaureate coursework that enables students to self-select into more challenging curricula (thus segregating academically focused students from their potentially less-diligent peers).

All public secondary schools in the USA are governed by local (usually city-wide) school boards that are generally overseen by state-level school boards. The federal government uses the new No Child Left Behind legislation and a few other techniques to encourage a small number of national standards.

Of the USA's 15+ million eligible students for the IOI, about 150–175 (0.001%) participate in USACO contests (47% in bronze division, 36% in silver division, 16% in gold division).

2. Task Development Background

The manageable number of students coupled with the natural fit of computer programming contests to the internet enables the USACO to offer six annual online programming contests (plus a qualification round) to USA students. Each contest has three divisions; each division contains 3–4 tasks, usually 9–10 tasks total. Annually, these contests consume 55–60 tasks.

The selection process thus garners a large amount of data for each student, so much that the student's worst performance can be dropped from statistics (most folks have a bad day once in a while) while still yielding meaningful means, trends, and so on. The monthly contests also enable tracking of students' improvement over time (and correlation with their participation in the USACO training site).

The USACO invites 16 (or so) of the best students to participate in the USA Invitational Computing Olympiad, an on-site selection contest and training week for the IOI. This camp has evolved to include six contests (four shorter, two longer) of 3–4 tasks each, an additional $\sim$20 tasks.

In addition, the USACO administers bonus contests (e.g., over the New Year holiday break) and special training events that also consume programming contest tasks.

All in all, the USACO creates and then consumes 75 or more fairly high-quality tasks per year, all developed by a volunteer coaching staff of about a dozen coaches located around the world.

The original implementation of the USACO grader enabled automated administration and grading of tasks in a contest environment. Task development and deployment proceeded manually, including manual setup and installation of tasks into the contest system.

Manual setup is well-known to be error-prone, and the USACO was no exception. Bugs and other issues were exacerbated by the just-in-time development effort, which often resulted in tasks (including text, solutions, and test data) being created the night before a contest (especially at camp with its very high rate of task consumption).

This development methodology not only created tremendous pressure and stress on the coaches but resulted in errors in the task statements, occasional tasks that were not as solvable as believed, test data that failed to meet the task specifications, and contest environment integration errors. Coupled with the stress and schedule pressure, the coaches felt the overhead of task development and contest integration was high.

3. Task Development Paradigms and Related Work

The existence of online, web-available, automated grading systems coupled with the advent of ever-cheaper hardware and widely-available reliable free software for development has fostered development of many grading systems around the world. Evolution of such systems usually proceeds from a web page that enables compilation and execution to a 'sandbox' to contain the behavior of submitted programs (imagine the security implementations of letting anyone send source code to run on your site). The compile/run system acquires a database of users in order to differentiate whose tasks are getting which results. The major issue of scaling comes into play as the abilities of a single CPU are exceeded.

Then the real challenge emerges: continuous development of new tasks and administration of contests on a schedule. Once scaling solutions and dedicated staff (volunteers, usually) are identified, the second-order issues can be attacked. Some country's organizations (e.g., those of Poland's Diks *et al.* (2008), Canada's Gordon Cormack, and The Netherlands' Tom Verhoeff (2008)) have devoted tremendous attention to the challenge of using black-box testing to ascertain the quality of submitted solutions to contest tasks.

The USACO's focus has been on providing a large, growing, and widely-available set of contests to the largest possible audience. While judging the quality of task ideas and text is difficult and extremely subjective, the works by the previously-named authors provide some guidance for creation and judging of test data. By the criteria of high-quality black-box testing, USACO's test data does not measure up to the standards of the IOI (and probably those of Diks, Cormack, and others). However, problem setter Richard Peng's efforts through the 2007–8 and 2008–9 seasons have increased test data quality dramatically. The data still meet the simple objective criterion: Do the test data enable fair and defensible differentiation among the contest competitors, preferably throughout the range of skills in a given division? For USACO, mostly because of the task-weighted scoring technique (also seen in use at IOI2008 in Egypt), both the top-, mid-, and low-level participants (in the Silver and Gold divisions) are differentiated quite well across a 1,000 point scale.

Diks *et al.* (2008) shared their Task Preparation Process at 2008's Olympiads in Informatics conference. Using their methodology and manpower, their tasks end up with superior 'model' solutions, better test development (black-box testing), and dramatically more complete and in-depth analyses when compared to USACO's ongoing efforts. Their series of training booklets is further evidence of their development of insightful and thorough solution analyses.

Verhoeff (2008), also at 2008's Olympiads in Informatics conference, shared his proposal for the Peach Exchange Format. Compared to this paper's work, it has a much more formal set of roles (and their descriptions) for developers, potentially much better 'background' (often mathematical) information for task development, 'notes' for developers (see Verhoeff, 2008; p. 202), and a formal specification for a file structure layout for task exchange.

USACO's paradigms more strongly emphasize throughput in the task development area at the expense of detailed analyses and extremely thorough black-box testing.

4. Requirements

One advantage of conducting a dozen IOI-level and IOI-size (for half of them) programming competitions per year is the rapid accrual of realistic statistics on the kinds of mistakes that creep into contest tasks. After one particularly stressful camp during which Senior USACO Coach Greg Galperin created a huge task matrix on a whiteboard to chart task development throughout the week, it became clear that automated scripts could not only aid in task development and debugging but also in contest administration system debugging. The decision was made to create a web-based task development system (which would not only integrate the scripts but also provide a distributed user interface). In hindsight, this is the next logical step in developing a contest administration framework once a grading system with a secure ‘sandbox’ is stable.

Experience made clear the requirements for tasks, which include rules like these:

- Text must be clear and complete.
- Text must be easily editable for repairs.
- Text must be easy to export to both web and to paper – since online contests use one method and on-site contests use the other.
- It is desirable for text to have multiple coaches’ ratings and comments – this helps determine which tasks belong on contests (vs. training or discard).
- Test data must be valid (i.e., conform to task’s data explanation).
- It must be easy to add, remove, reorder test data.
- The system must ease creation of data validators (Verhoeff, 2000) – the data validator was one of the most effective schemes for eliminating complaints and regrades.
- It is desirable to create data validator mechanically – the ability to have a program read a task and deduce the input data format ensures a uniform, proper presentation of input requirements in addition to saving time.
- The system must support all styles of IOI tasks (simple answer, multiple answer, reactive, output-only, programmatic grader, etc.).
- The system must enable configuration of submission feedback (categories like: case is required to be correct for submission to proceed, report right/wrong for one or a set of tasks, or no feedback at all).
- The system must enable configuration of multiple-tests per case.
- Task must be solvable within the contest constraints.
- The system must provide automated evaluation of proposed solutions in contest environment (running in any other environment does not “prove” solvability).
- The system must provide comparison of answers from various solutions/solvers – this must be mechanical since answer sets can be large and similar.
- It must be easy to enter answer-graders and format-checkers.

Requirements for task development include:

– easy navigation,
– ability to create pool of tasks to use in various contests,
– ability to see status (including coach ratings) of each task,
– ability to see status of each contest,

- fast navigation for rating many tasks – fast, easy navigation encourages coaches to rate more tasks,
- convenient contest setup and parameter management,
- archiving scheme so tasks can smoothly move to training or other places – one-click archiving makes for easy reuse of tasks,
- collection of task analysis text,
- ease of set up and take down for contests, including automatic start/stop.

5. Chosen Task Development Paradigms

The task development paradigm centers on the task as a ‘unit’ with this (expandable) list of components:

- numerical problem ID,
- task author,
- task analysis status,
- task editing status,
- count and list of task solutions (and solvers) with solution-agreement status,
- data validator and its status,
- test data and its feedback type,
- task ratings,
- task algorithm,
- estimated time for ‘ideal student’ to solve,
- task abbreviation/short name,
- task title,
- task text.

The task’s text includes these items (which are stored and manipulated as separate entities):

- full task name,
- short task name/abbreviation,
- assigned owner for task,
- presentation order,
- division,
- author,
- date,
- text body,
- input format (including broken-out start/stop line numbers),
- sample input,
- input explanation,
- output format (including broken-out line numbers),
- sample output,
- output explanation,
- the task’s test data representation includes:

- each of the test cases,
- memory limit,
- default time limit,
- grader program for task output (optional),
- format checker for task output (optional),
- auxiliary filename and contents (e.g., for external dictionary),
- data validator program,
- scoring tables for aggregates (when score depends on more than one test case),
- per-test case time limit.

Notes for test data are consolidated with notes in the task discussion.

Additionally, the development system's paradigm includes the ability to set new task-development instances (for training tasks, university training/homework tasks, personal task development gateways, etc.).

6. Current Implementation

The system supplies several web pages to manipulate the task's components:

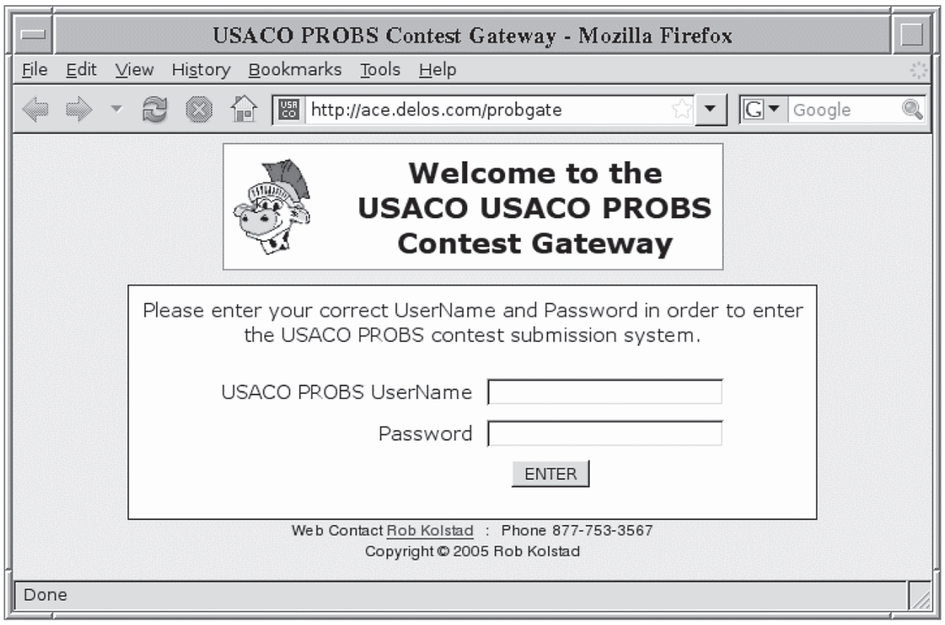
- the login page,
- the main status and navigation page,
- the text presentation, ratings, algorithm, solution time, ratings, and comments page,
- the problem solution page,
- the test data management page,
- the contest management page.

Subsections below detail each of these.

The task development infrastructure is part of an integrated training/contest administration system written almost entirely in the perl programming language (the 'sandbox' being the only exceptional program; C felt like a better tool for its implementation). About two dozen scripts (including administrative programs) total about 10,000 lines of perl for the contest development infrastructure. At the time of implementation, perl was one of only a few implementation tools mature enough to implement the system effectively. Today, one could probably use any of several languages and/or tools. The author is an expert perl programmer, so it was a natural choice at the time. The system runs on FreeBSD and Linux (with a migration to a pure Linux environment planned for the near future).

7. Login/Authentication

The current implementation of the task development system is accessed via a login page on the web; see Fig. 1. A database keeps track of both coaches and competitors' usernames and passwords (over 105K as of 1 May 2009).


USACO PROBS Contest Gateway - Mozilla Firefox

File Edit View History Bookmarks Tools Help
http://ace.delos.com/probgate
Google

Welcome to the
USACO USACO PROBS
Contest Gateway

Please enter your correct UserName and Password in order to enter the USACO PROBS contest submission system.

USACO PROBS UserName

Password

Web Contact Rob Kolstad : Phone 877-753-3567
Copyright © 2005 Rob Kolstad

Fig. 1. Authentication page for problem editing gateway.

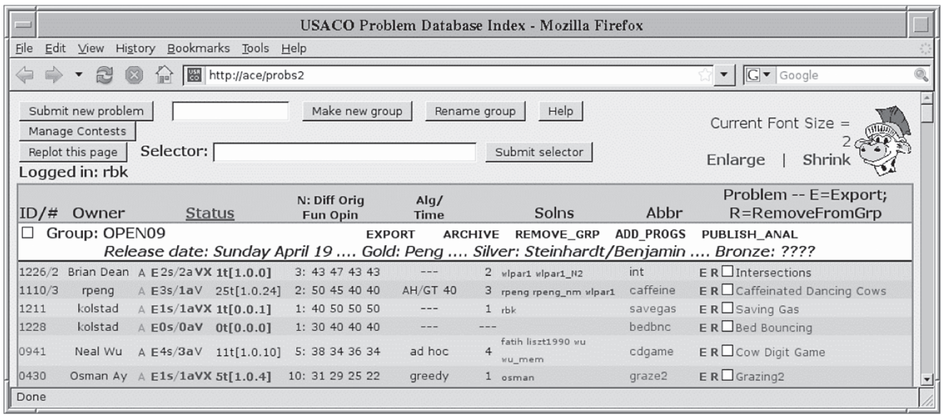
8. Status and Navigation

The main status and navigation page shows all the managed tasks in groups that have been assigned by the coaches. Typical groupings are for upcoming contests, unassigned tasks of a similar type, and unassigned tasks of a given division (e.g., gold, silver, or bronze).

Fig. 2 shows a small fraction of the top of the main status navigation page. The very top includes some submit buttons used for group naming and other functions:

- Submit new problem: plots a task input page that collects each of the sections mentioned above for a given task.
- Make new group: creates a new group with the name supplied in the input box.
- Rename group: Changes the name of the group whose checkbox has been clicked to the name supplied in the input box.
- Help: plots a page of explanatory text.
- Manage Contests: brings up the contest setup page.
- Replot this page: redisplays the page with all updates.
- Submit selector: activates a filter that displays only those groups whose name matches the regular expression entered in the Selector box (dramatically reduces traffic when working on a single contest).
- Enlarge and Shrink: Increases or decreases the typeface size.

The headings show up next followed by the first (of many) group headers. The group header rows include the header (and its checkbox) with the group name and some commands in addition to a single line status message previously entered by a contest director.



Logged in: rbk

ID/#	Owner	Status	N: Diff Orig Fun Opin	Alg/ Time	Solns	Abbr	Problem -- E=Export; R=RemoveFromGrp
☐ Group: OPEN09			EXPORT	ARCHIVE	REMOVE_GRP	ADD_PROGS	PUBLISH_ANAL
Release date: Sunday April 19 .... Gold: Peng .... Silver: Steinhardt/Benjamin .... Bronze: ????							
1226/2	Brian Dean	A E2s/2aVX 1t[1.0.0]	3: 43 47 43 43	---	2 wlpar1 wlpar1_N2	int	E R ☐ Intersections
1110/3	rpeng	A E3s/1aV 25t[1.0.24]	2: 50 45 40 40	AH/GT 40	3 rpeng rpeng_nm wlpar1	caffeine	E R ☐ Caffeinated Dancing Cows
1211	kolstad	A E1s/1aVX 1t[0.0.1]	1: 40 50 50 50	---	1 rbk	savegas	E R ☐ Saving Gas
1228	kolstad	A E0s/0aV 0t[0.0.0]	1: 30 40 40 40	---	---	bedbnc	E R ☐ Bed Bouncing
0941	Neal Wu	A E4s/3aV 11t[1.0.10]	5: 38 34 36 34	ad hoc	4 fatih liszt1990 wu wu_mem	cdgame	E R ☐ Cow Digit Game
0430	Osman Ay	A E1s/1aVX 5t[1.0.4]	10: 31 29 25 22	greedy	1 osman	graze2	E R ☐ Grazing2

Fig. 2. Status and navigation page.

The per-group commands include:

- EXPORT – Copy this group's tasks to the contest whose name matches the group name.
- ARCHIVE – Copy this group's tasks to the training task development system and remove them from this page.
- REMOVE_GRP – Move the tasks in this group to the section below for unallocated tasks and delete the group name and its status message.
- ADD_PROBS – Move the tasks whose checkbox has been marked into this group.
- PUBLISH_ANALYSIS – Retrieve the analyses entered for this group's tasks and copy them to the web for display with task data.
- ADD_NOTE – Shown only for special users when the contest 'note' is empty; bring up note-text editing page.

The rest of the page shows the name and status for each of the tasks managed by this particular task management system (in this case, 188 tasks are summarized below). Each line contains a set of color-coded information about this task:

- Task Number: The numerical ID of the task. Clicking the number displays the text-editing page.
- Task Number/Order: The presentation order number of the task, if it exists, is displayed after the task.
- Owner: The next column shows the task's owner.
- Status: A: The color of the letter A indicates the existence of the analysis (red for missing, blue for present). Clicking the A brings up the analysis entry page.
- Status: E: The color of the letter E indicates whether the task has been final-edited (bold-blue for not-edited, black for edited).
- Status: Xs/Ya: X is the number of solutions submitted; Y is the number of solutions in agreement with the master solution. Bold-blue indicates insufficient solutions or agreements. Clicking this item brings up the solution submission and status page.

- Status: VX: Bold-blue V means the validator is not submitted; presence of the bold-blue X indicates the existence of test data that does not pass the validator's tests.
- Status: Xt[A.B.C]: X is the number of test cases that have been submitted. A is the number of tests required to be passed for submission to succeed; B is the number of tests whose results will be shown when a submission is made; C is the number of tests that will be run normally. Clicking this item brings up the test data submission page.
- N: Diff Orig Fun Opin: These five items represent respectively the number of ratings entered by coaches as their evaluation of the task: the average difficulty rating, the average originality rating, the average fun rating, and the average overall opinion. The integer averages are the product of ten and the mean of all the 1–5 rating entries.
- Alg/Time: These two items show the algorithm abbreviation (as entered on the problem display page by a knowledgeable coach) and the ideal student's solution time in minutes (also entered on the problem display page).
- Solns: The number represents the number of solution programs submitted; the subsequent login IDs are the names of the solutions submitted (or, sometimes, names assigned by the submitter).
- Abbr: The short problem name, its abbreviation.
- E: Clicking the E exports the single task to the contest administration system.
- R: Clicking the R removes the task from its group and moves it to the unassigned task list at the bottom of the page.
- Name: The long name (title) of the task.

Problem creation and contest configuration proceed in parallel. To set up a contest, a coach enters a canonical name (e.g., NOV09, DEC09, JAN10, etc.) and clicks **Make new group**. He then ticks the checkboxes of tasks to move to that group and clicks the **ADD_PROGS** tag on the group's header.

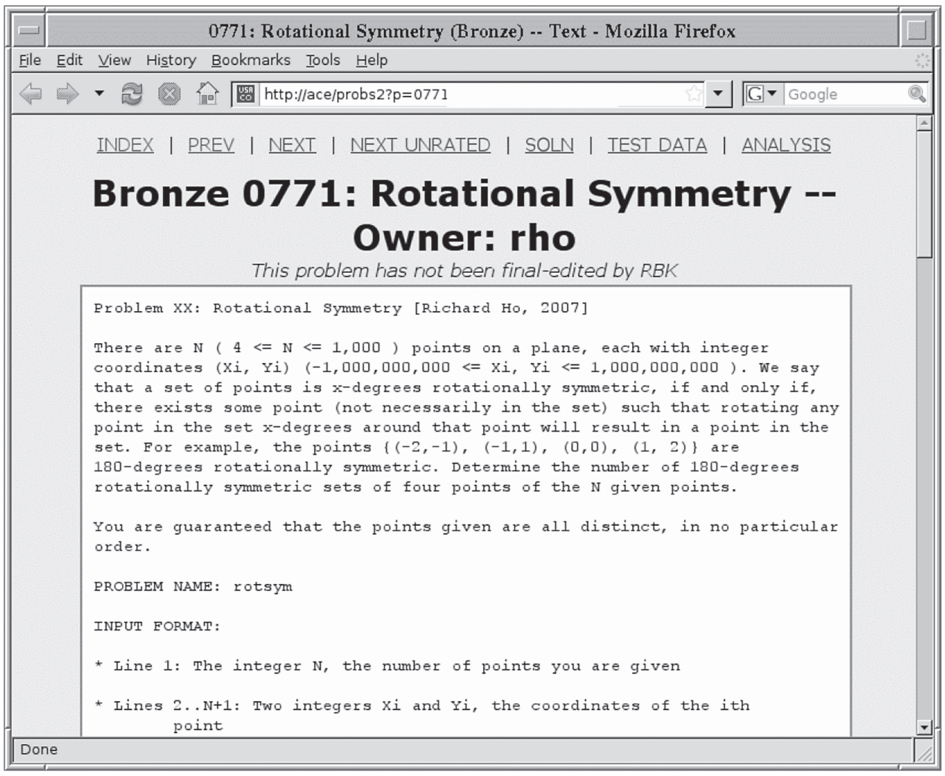
Coaches can read the task's text, enter ratings, enter solutions, enter test data, add a validator, and submit output format checkers and graders. Any replot of the status change shows all updates that have been entered.

When a problem is almost "ready" (i.e., it is final-edited, has at least two solutions that agree with the master solution, has a validator that passes all the input data, and has at least eight test cases), the background color around the "Status" fields changes to light green. When the analysis is entered (generally not required until the contest is almost over), the background shade changes to dark green.

A contest is fully ready when a dark green stripe runs completely through the "Status" column's fields.

9. Text, Comments, and Ratings

Fig. 3a shows the top part of a typical problem presentation page. It leads off with navigation links and then prominently displays the problem's name and owner (and also


0771: Rotational Symmetry (Bronze) -- Text - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://ace/probs2?p=0771
Google
INDEX | PREV | NEXT | NEXT UNRATED | SOLN | TEST DATA | ANALYSIS
Done

Bronze 0771: Rotational Symmetry -- Owner: rho

This problem has not been final-edited by RBK

```
Problem XX: Rotational Symmetry [Richard Ho, 2007]

There are N ( 4 <= N <= 1,000 ) points on a plane, each with integer
coordinates (Xi, Yi) (-1,000,000,000 <= Xi, Yi <= 1,000,000,000 ). We say
that a set of points is x-degrees rotationally symmetric, if and only if,
there exists some point (not necessarily in the set) such that rotating any
point in the set x-degrees around that point will result in a point in the
set. For example, the points {(-2,-1), (-1,1), (0,0), (1, 2)} are
180-degrees rotationally symmetric. Determine the number of 180-degrees
rotationally symmetric sets of four points of the N given points.

You are guaranteed that the points given are all distinct, in no particular
order.

PROBLEM NAME: rotsym

INPUT FORMAT:

* Line 1: The integer N, the number of points you are given

* Lines 2..N+1: Two integers Xi and Yi, the coordinates of the ith
        point
```

Fig. 3a. Top of text presentation page.

algorithm and solution time, if available). The editing status appears in italics just below the task name. The problem text itself then appears.

The problem text includes many components. The largest component is usually the problem statement itself which, by USACO convention, includes the limits on all the various input parameters.

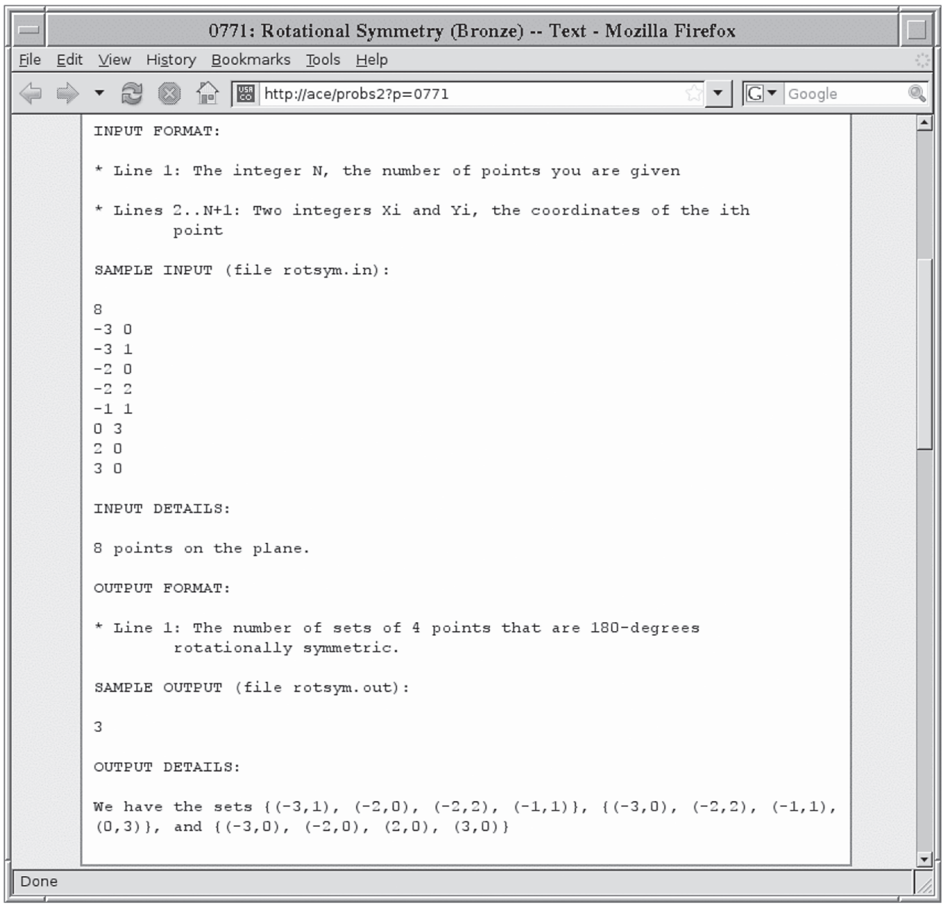
Following the problem text, the problem's short name (abbreviation) appears with an easy-to-find capitalized heading.

Fig. 3b shows the input/output section of this task's display. The input format still uses the old-fashioned "Line 1, Line 2, ..." notation instead of the more modern "Next 3 lines..." notation. The starting line and optional ending line are entered for each "section" of input as part of the input process. These fields are used by the mechanical validator creation routine to create loops that check the input.

The "Input details" section is optional, as is "Output Details."

Fig. 3c shows the next set of information from the text page. It includes navigation buttons to:

– edit the problem's text, name, division, or input/output parameters,
– show recent changes to the text,
– navigate to the test data or solution pages,


0771: Rotational Symmetry (Bronze) -- Text - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://ace/probs2?p=0771
Google
Done

```
INPUT FORMAT:

* Line 1: The integer N, the number of points you are given

* Lines 2..N+1: Two integers Xi and Yi, the coordinates of the ith
        point

SAMPLE INPUT (file rotsym.in):

8
-3 0
-3 1
-2 0
-2 2
-1 1
0 3
2 0
3 0

INPUT DETAILS:

8 points on the plane.

OUTPUT FORMAT:

* Line 1: The number of sets of 4 points that are 180-degrees
        rotationally symmetric.

SAMPLE OUTPUT (file rotsym.out):

3

OUTPUT DETAILS:

We have the sets {(-3,1), (-2,0), (-2,2), (-1,1)}, {(-3,0), (-2,2), (-1,1),
(0,3)}, and {(-3,0), (-2,0), (2,0), (3,0)}
```

Fig. 3b. End of text presentation on text presentation page.

– remove the task altogether,
– remove all entered data except the task's text,
– mark the task's editing as approved (only available to special directors),
– a shortcut to submit a solution (presumably created while reading the text above).

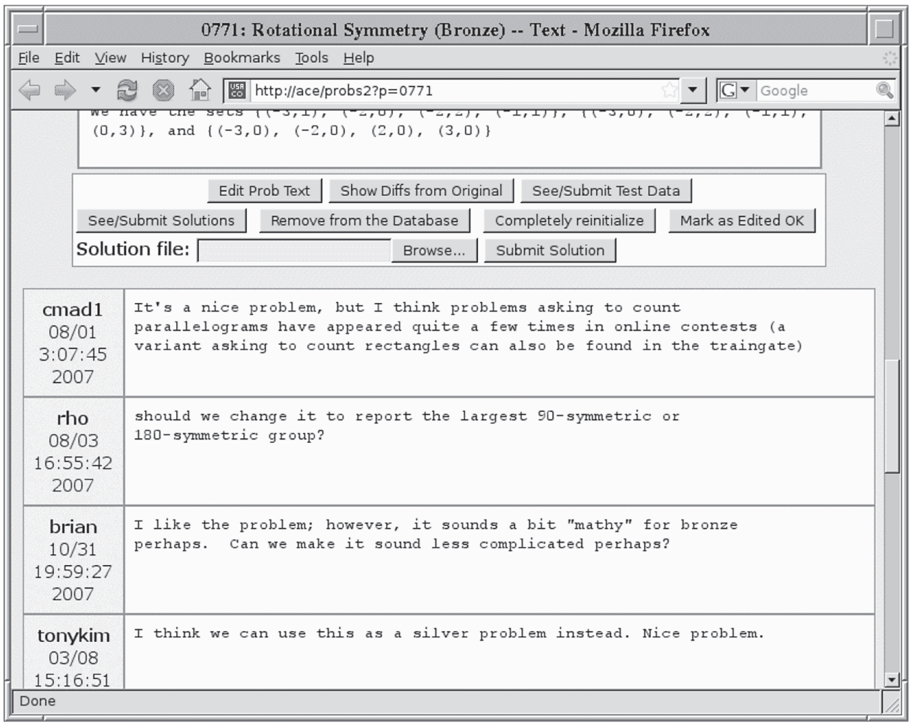
Finally, Fig. 3d shows the rating and comment box along with the coaches' ratings. Some coaches are empowered to enter an estimated "Solution Time" and "Solution Algorithm" which then appear on the summary page. When a large pool of tasks is available, a coach can scan solution times and algorithms to create a contest very quickly.

10. Text Editing

The text editing page is mostly self-explanatory (see Fig. 4a) with fields to enter:

– the full task name,

•



```
(0,3)}, and {(-3,0), (-2,0), (2,0), (3,0)}
```

cmad1 08/01 3:07:45 2007	It's a nice problem, but I think problems asking to count parallelograms have appeared quite a few times in online contests (a variant asking to count rectangles can also be found in the traingate)
rho 08/03 16:55:42 2007	should we change it to report the largest 90-symmetric or 180-symmetric group?
brian 10/31 19:59:27 2007	I like the problem; however, it sounds a bit "mathy" for bronze perhaps. Can we make it sound less complicated perhaps?
tonykim 03/08 15:16:51	I think we can use this as a silver problem instead. Nice problem.

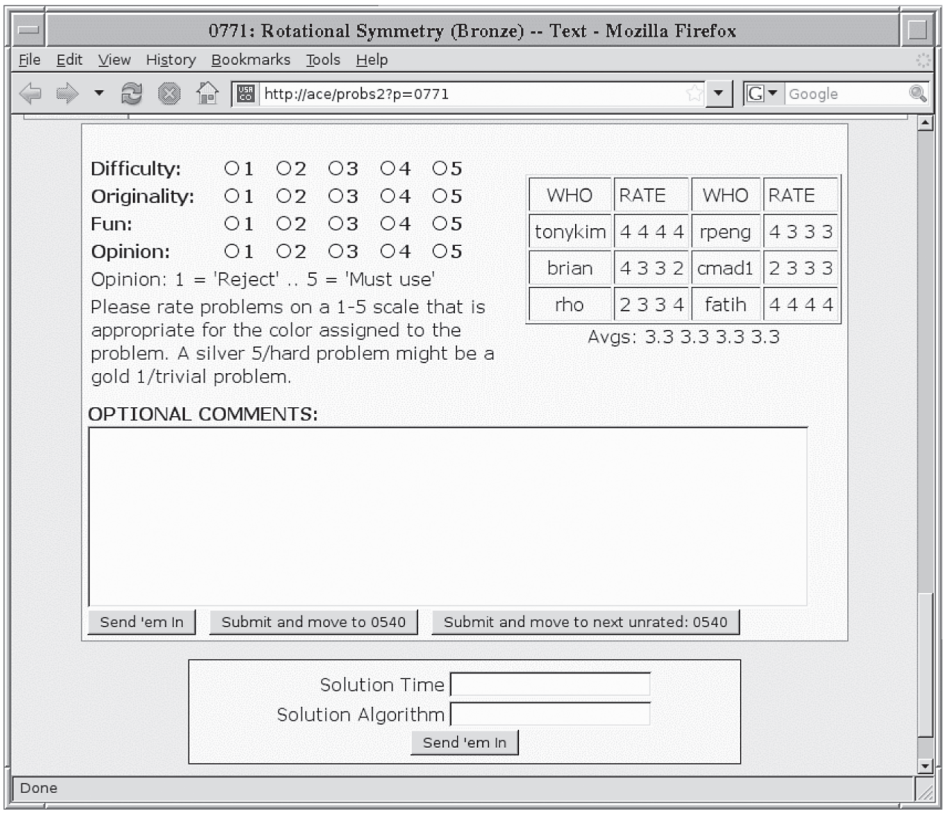
Fig. 3c. Text page navigation and comments.

– the short task name (abbreviation),
– the task's owner,
– the optional task presentation order,
– the task's division,
– the task's author,
– the task's date,
– when the task was used,
– the task's text.

The page's display then continues with the self-explanatory input and output specifications; see Fig. 4b. This figure has been edited to remove blank fields and white space. Terminology has evolved over time; the "Input Explanation" is displayed to competitors as "Input Details."

11. Solution Page

Coaches spend most of their time on the "Solutions Page" honing their solutions and ensuring the answers are correct. The page enables saving of multiple solutions for each coach so they can compare various algorithms and techniques.


0771: Rotational Symmetry (Bronze) -- Text - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://ace/probs2?p=0771

Difficulty: ○1 ○2 ○3 ○4 ○5
Originality: ○1 ○2 ○3 ○4 ○5
Fun: ○1 ○2 ○3 ○4 ○5
Opinion: ○1 ○2 ○3 ○4 ○5
Opinion: 1 = 'Reject' .. 5 = 'Must use'
Please rate problems on a 1-5 scale that is appropriate for the color assigned to the problem. A silver 5/hard problem might be a gold 1/trivial problem.

WHO	RATE	WHO	RATE
tonykim	4 4 4 4	rpeng	4 3 3 3
brian	4 3 3 2	cmad1	2 3 3 3
rho	2 3 3 4	fatih	4 4 4 4

Avgs: 3.3 3.3 3.3 3.3

OPTIONAL COMMENTS:

Send 'em In | Submit and move to 0540 | Submit and move to next unrated: 0540

Solution Time
Solution Algorithm
Send 'em In

Done

Fig. 3d. Ratings, comment box, and rating navigation.

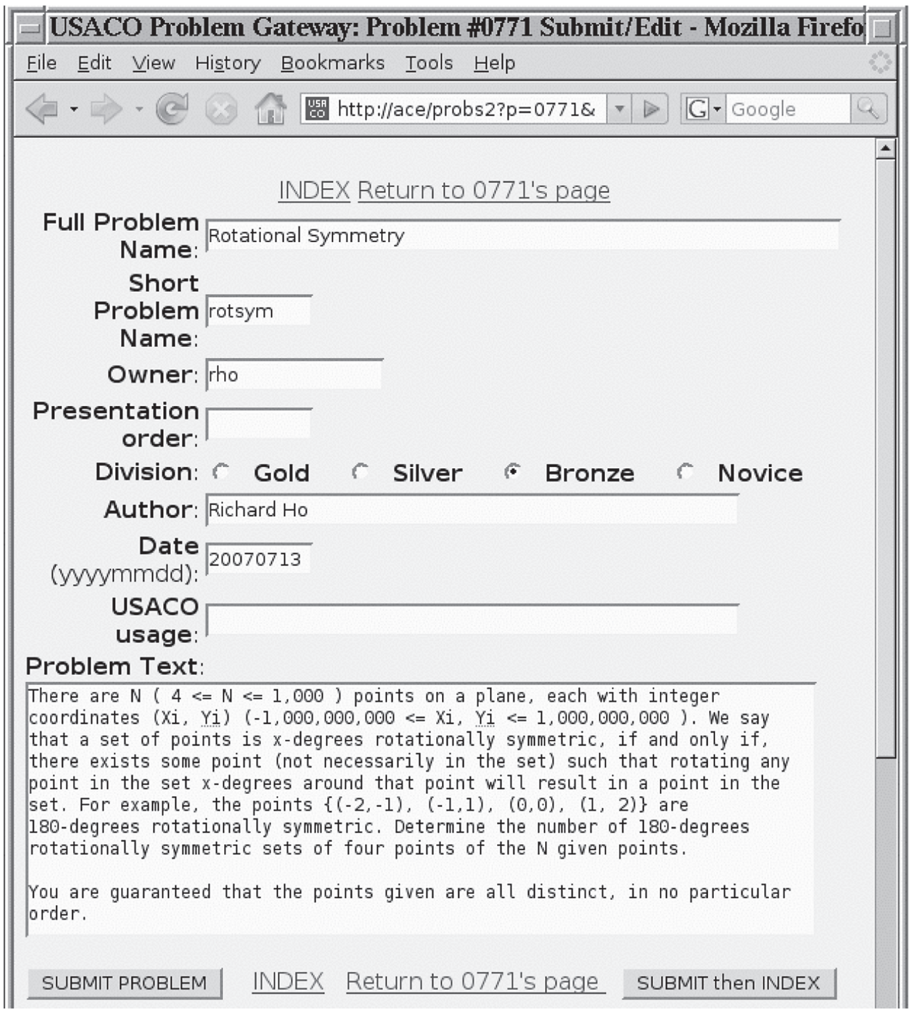
Fig. 5 shows a typical solution page. The top line provides navigation. The table shows the status of all submitted solutions:

- Solver is the solution's name, most often that of the submitter.
- Size shows the length of the solution (or the number of cases in the official solution).
- Available displays which cases have been run (and, in the case of disagreeing answers, red denotes those that don't agree).
- Actions include deleting the solution, (re-)running it, copying it to the official solution (along with its answers), or 'X' to mark the available case solutions as invalid (in order to re-run them).

Auxiliary comparison buttons enable selected runs to be displayed either in full or as 'diff's. One can run or re-run all solutions.

The **Solution Agreement Table** displays smiling (or not-so-smiling) faces to display which programs are in agreement. The faces are clickable to show the differences in the output files. New, unrun test cases display as gray numbers in the 'Available' column; the Solution Agreement Table displays question marks until the solutions are run on the new data (generally with a single click to "Run all Solutions").

•



USACO Problem Gateway: Problem #0771 Submit/Edit - Mozilla Firefo

File Edit View History Bookmarks Tools Help
http://ace/probs2?p=0771&
Google

INDEX Return to 0771's page

Full Problem Name:	Rotational Symmetry
Short Problem Name:	rotsym
Owner:	rho
Presentation order:	
Division:	Gold Silver [✓ Bronze] Novice
Author:	Richard Ho
Date (yyyymmdd):	20070713
USACO usage:	

Problem Text:

```
There are N ( 4 <= N <= 1,000 ) points on a plane, each with integer
coordinates (Xi, Yi) (-1,000,000,000 <= Xi, Yi <= 1,000,000,000 ). We say
that a set of points is x-degrees rotationally symmetric, if and only if,
there exists some point (not necessarily in the set) such that rotating any
point in the set x-degrees around that point will result in a point in the
set. For example, the points {(-2,-1), (-1,1), (0,0), (1, 2)} are
180-degrees rotationally symmetric. Determine the number of 180-degrees
rotationally symmetric sets of four points of the N given points.

You are guaranteed that the points given are all distinct, in no particular
order.
```

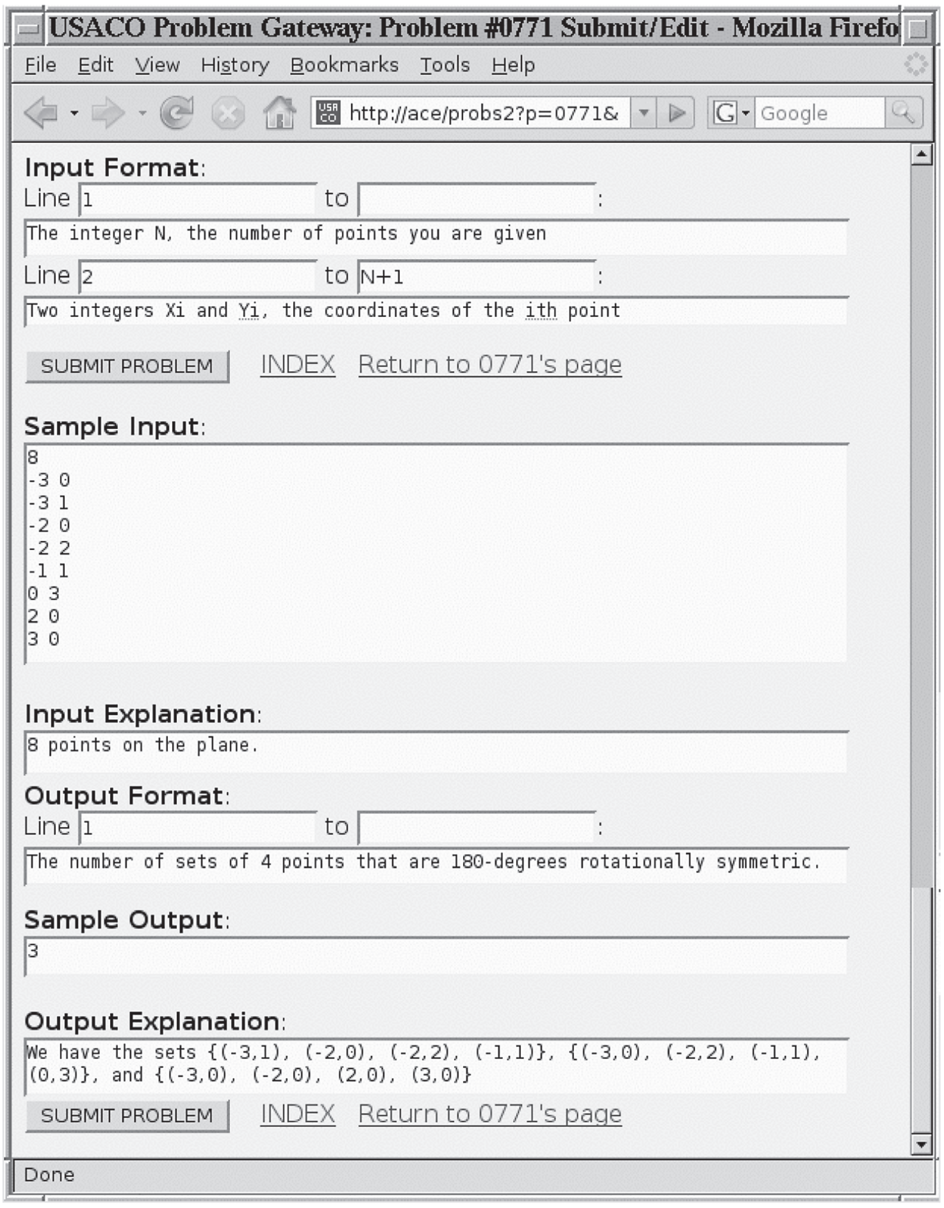
Fig. 4a. Text editing: parameters and body.

The **Best CPU Time per Case** table is self-explanatory and is used to ensure that tasks have plenty of "head room" for slightly inferior algorithms or implementations – or the effect of other programming languages (most notably, Java).

Finally, submission boxes enable sending in of new solutions and renaming existing solutions.

12. Test Data

The test data submission page handles not only the submission and display of test data but also (see Fig. 6a):


USACO Problem Gateway: Problem #0771 Submit/Edit - Mozilla Firefo
File Edit View History Bookmarks Tools Help
http://ace/probs2?p=0771&
Google

Input Format:

Line 1 to :

```
The integer N, the number of points you are given
```

Line 2 to N+1 :

```
Two integers Xi and Yi, the coordinates of the ith point
```

SUBMIT PROBLEM INDEX Return to 0771's page

Sample Input:

```
8
-3 0
-3 1
-2 0
-2 2
-1 1
0 3
2 0
3 0
```

Input Explanation:

```
8 points on the plane.
```

Output Format:

Line 1 to :

```
The number of sets of 4 points that are 180-degrees rotationally symmetric.
```

Sample Output:

```
3
```

Output Explanation:

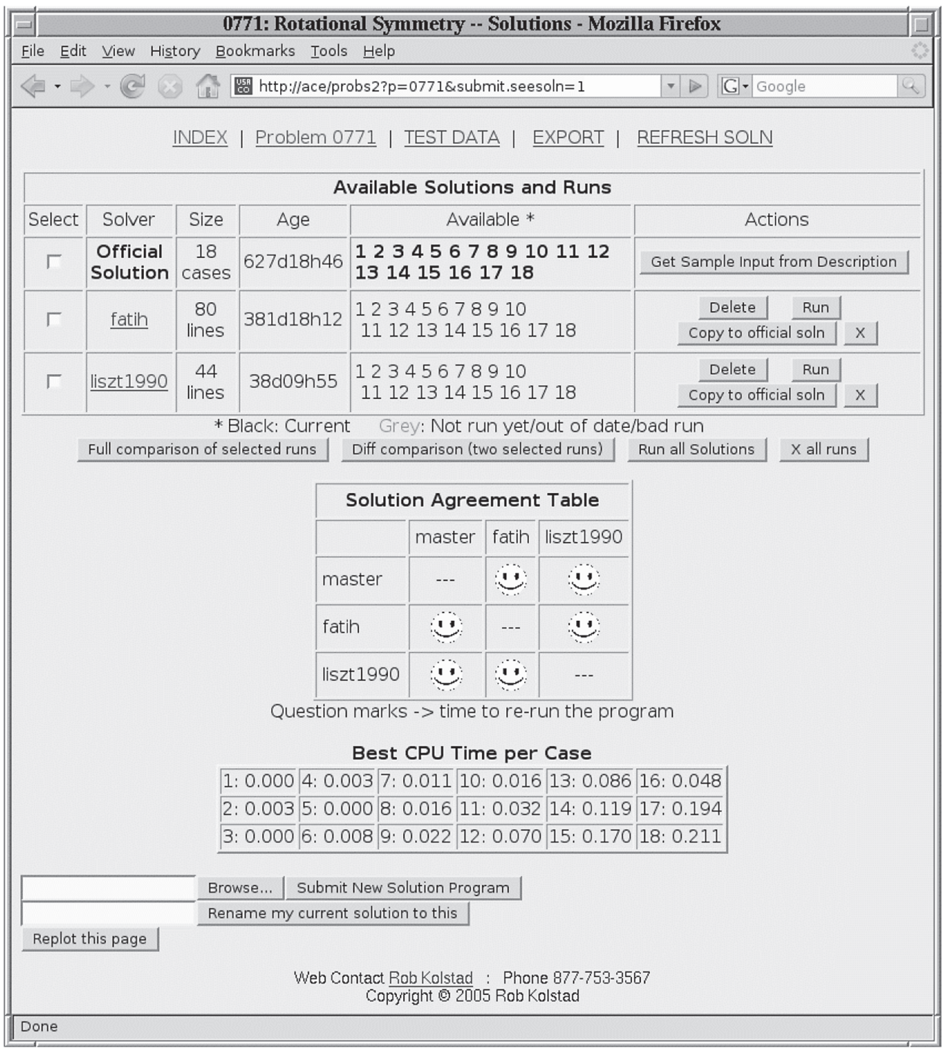
```
We have the sets {(-3,1), (-2,0), (-2,2), (-1,1)}, {(-3,0), (-2,2), (-1,1),
(0,3)}, and {(-3,0), (-2,0), (2,0), (3,0)}
```

SUBMIT PROBLEM INDEX Return to 0771's page

Done

Fig. 4b. Input and output specifications.

- memory limit,
- default time limit (optionally overridden on a case-by-case basis),
- output grader program uploading,
- format checker program uploading,
- auxiliary filename and actual file contents uploading,



0771: Rotational Symmetry -- Solutions - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://ace/probs2?p=0771&submit.seesoln=1

INDEX | Problem 0771 | TEST DATA | EXPORT | REFRESH SOLN

Available Solutions and Runs

Select	Solver	Size	Age	Available *	Actions
☐	**Official Solution**	18 cases	627d18h46	**1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18**	Get Sample Input from Description
☐	fatih	80 lines	381d18h12	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18	Delete Run Copy to official soln X
☐	liszt1990	44 lines	38d09h55	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18	Delete Run Copy to official soln X

* Black: Current Grey: Not run yet/out of date/bad run

Full comparison of selected runs | Diff comparison (two selected runs) | Run all Solutions | X all runs

Solution Agreement Table

	master	fatih	liszt1990
master	---	☺	☺
fatih	☺	---	☺
liszt1990	☺	☺	---

Question marks -> time to re-run the program

Best CPU Time per Case

1: 0.000	4: 0.003	7: 0.011	10: 0.016	13: 0.086	16: 0.048
2: 0.003	5: 0.000	8: 0.016	11: 0.032	14: 0.119	17: 0.194
3: 0.000	6: 0.008	9: 0.022	12: 0.070	15: 0.170	18: 0.211

Browse... Submit New Solution Program

Rename my current solution to this

Replot this page

Web Contact Rob Kolstad : Phone 877-753-3567
Copyright © 2005 Rob Kolstad

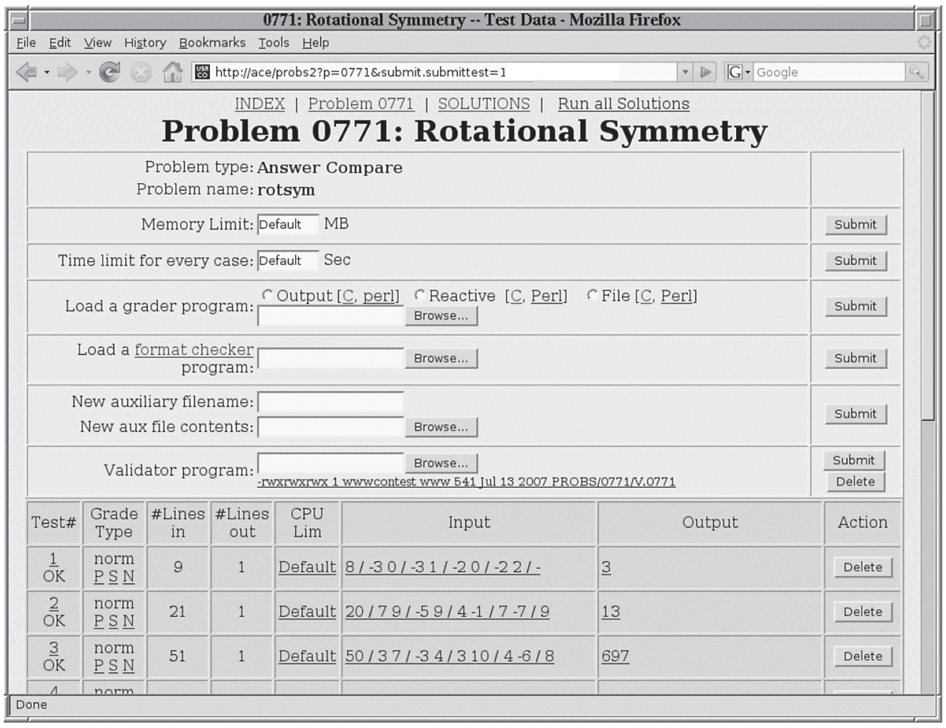
Done

Fig. 5. Solution submission and display page.

- validator uploading.

The test data display includes:

- Test#: The case number of this test (not always sequential in early stages of test data development).
- Grade Type: Whether this test case is run upon submission (with or without a requirement that the test be passed for successful submission) or later for results.
- #Lines in and Lines out: size of the test case.
- CPU Lim: The override value for the maximum CPU time.
- Input and Output: summaries of the first lines of input and output for the test case.



0771: Rotational Symmetry -- Test Data - Mozilla Firefox

http://ace/probs2?p=0771&submit.submittest=1

INDEX | Problem 0771 | SOLUTIONS | Run all Solutions

Problem 0771: Rotational Symmetry

Problem type: Answer Compare
Problem name: rotsym

Memory Limit: Default MB

Time limit for every case: Default Sec

Load a grader program: Output [C, perl] Reactive [C, Perl] File [C, Perl]

Load a format checker program:

New auxiliary filename:
New aux file contents:

Validator program:
-rwxrwxrwx 1 wwwcontest www 541 Jul 13 2007 PROBS/0771/V.0771

Test#	Grade Type	#Lines in	#Lines out	CPU Lim	Input	Output	Action
1 OK	norm P S N	9	1	Default	8 / -3 0 / -3 1 / -2 0 / -2 2 / -	3	Delete
2 OK	norm P S N	21	1	Default	20 / 7 9 / -5 9 / 4 -1 / 7 -7 / 9	13	Delete
3 OK	norm P S N	51	1	Default	50 / 3 7 / -3 4 / 3 10 / 4 -6 / 8	697	Delete

Done

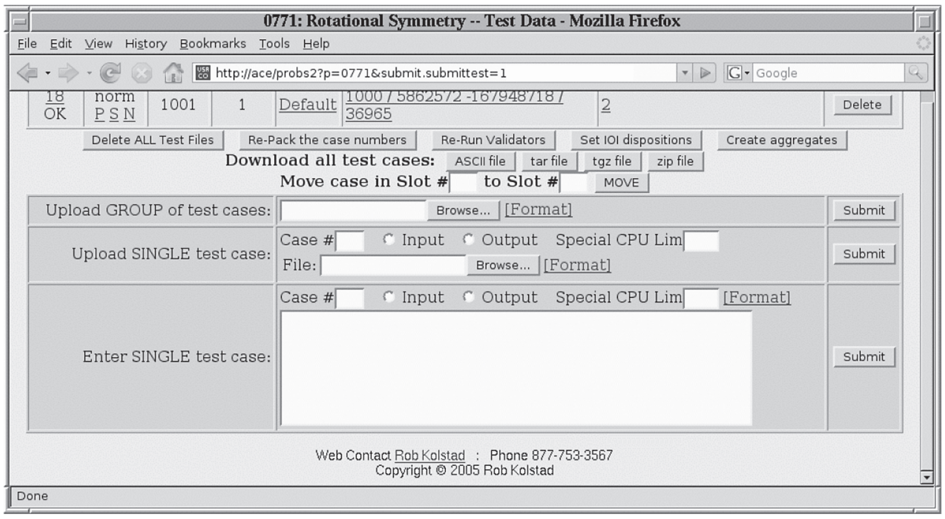
Fig. 6a. Test data programs and display.

- Action: A button to delete the test case.

The bottom half of the test data submission page (see Fig. 6b) includes:

- Delete ALL Test Files: The big gun.
- Re-Pack the case numbers: To make the case numbers be sequential (often used after test case deletion).
- Re-Run Validators: Does the obvious thing. This is handy when someone is editing the validator in the filesystem instead of via the web page.
- Set IOI dispositions: Obsolete functionality that marked every other case as "show result during submission."
- Create aggregates: Invokes functionality that enables combining of individual test cases into super-cases.
- Download all test cases: Creates any of four portable formats for moving all the test cases around (or editing) in a single file (which is reloadable below).
- Move case ...: Self-explanatory.

The bottom of the page is where the actual uploading or entry of test cases occurs. The fields are self-explanatory.


0771: Rotational Symmetry -- Test Data - Mozilla Firefox

http://ace/probs2?p=0771&submit.submittest=1

18 OK	norm P S N	1001	1	Default	1000 / 5862572 -1679487187 / 36965	2	Delete

Delete ALL Test Files | Re-Pack the case numbers | Re-Run Validators | Set IOI dispositions | Create aggregates

Download all test cases: ASCII file | tar file | tgz file | zip file

Move case in Slot # to Slot # MOVE

Upload GROUP of test cases:	Browse... [Format]	Submit
Upload SINGLE test case:	Case # Input Output Special CPU Lim File: Browse... [Format]	Submit
Enter SINGLE test case:	Case # Input Output Special CPU Lim [Format]	Submit

Web Contact Rob Kolstad : Phone 877-753-3567
Copyright © 2005 Rob Kolstad

Done

Fig. 6b. Test data submission.

13. Contest Management

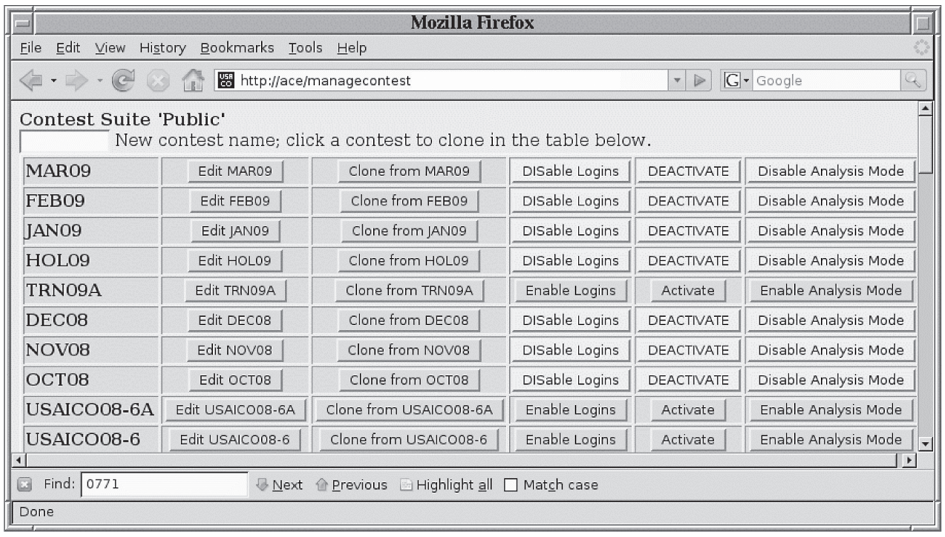
The contest management page (see Fig. 7a) has a relatively boring presentation but is, of course, the center of administrative power. It uses the paradigm of cloning old contests to create new ones (modifying the ioiconfig.perl file from one contest to update it with the new contest's information). Each contest is stored in a contest directory with its configuration, problem texts, submissions, and so on. Functionality includes:

- Edit: displays the configuration editing page (see below).
- Clone: Creates a new contest with the name entered in the box near the top by cloning a previous contest.
- Enable logins/DISable Logins: Toggles the configuration to allow non-contest-directors to login or not.
- Activate/DEACTIVATE: Manipulates a table that shows which contests are available for logging in. Without available login, participants cannot even see information about the contest and cannot login.
- Enable Analysis Mode/Disable Analysis Mode: Toggle the configuration file variable that specifies whether the contest is analysis mode (vs. regular competition mode).

14. Contest Configuration

The contest configuration page enables input for the all the standard contest configuration parameters:

– contest name,



Contest Suite 'Public'
New contest name; click a contest to clone in the table below.

MAR09	Edit MAR09	Clone from MAR09	DISable Logins	DEACTIVATE	Disable Analysis Mode
FEB09	Edit FEB09	Clone from FEB09	DISable Logins	DEACTIVATE	Disable Analysis Mode
JAN09	Edit JAN09	Clone from JAN09	DISable Logins	DEACTIVATE	Disable Analysis Mode
HOL09	Edit HOL09	Clone from HOL09	DISable Logins	DEACTIVATE	Disable Analysis Mode
TRN09A	Edit TRN09A	Clone from TRN09A	Enable Logins	Activate	Enable Analysis Mode
DEC08	Edit DEC08	Clone from DEC08	DISable Logins	DEACTIVATE	Disable Analysis Mode
NOV08	Edit NOV08	Clone from NOV08	DISable Logins	DEACTIVATE	Disable Analysis Mode
OCT08	Edit OCT08	Clone from OCT08	DISable Logins	DEACTIVATE	Disable Analysis Mode
USAICO08-6A	Edit USAICO08-6A	Clone from USAICO08-6A	Enable Logins	Activate	Enable Analysis Mode
USAICO08-6	Edit USAICO08-6	Clone from USAICO08-6	Enable Logins	Activate	Enable Analysis Mode

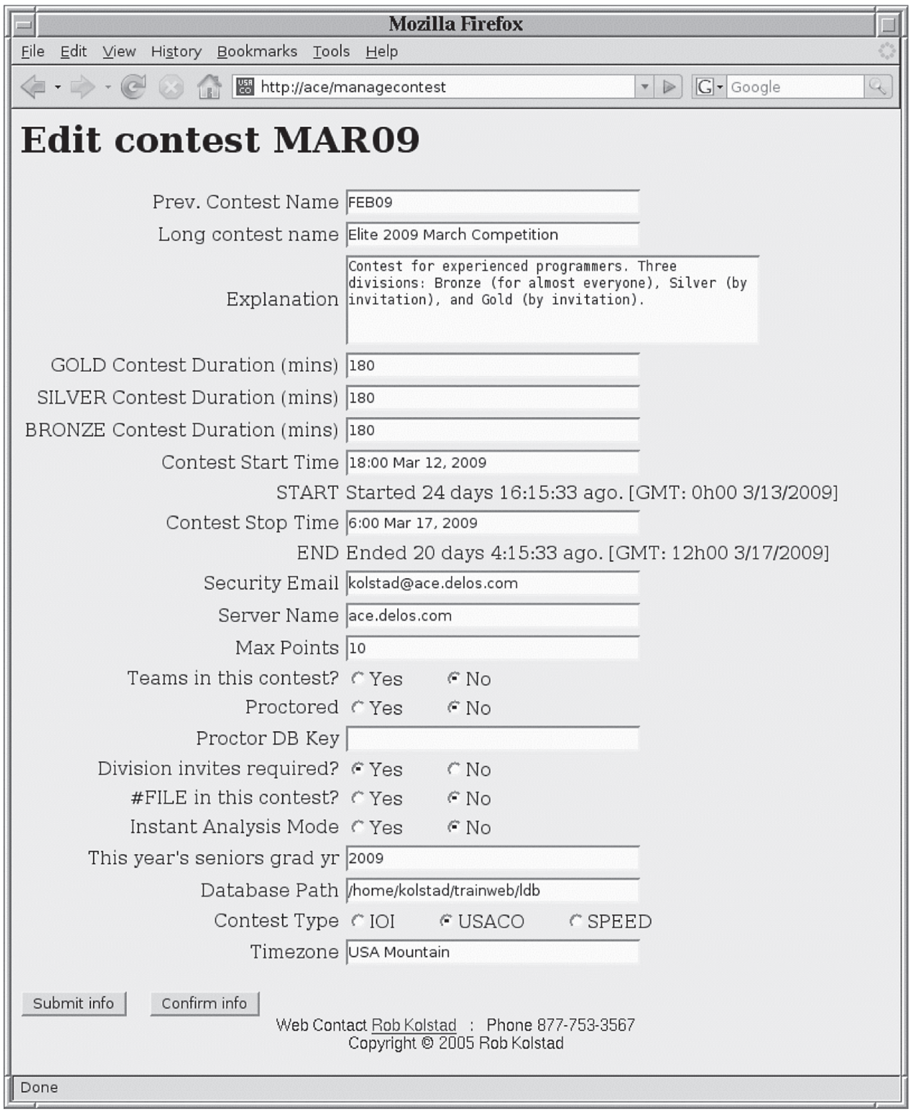
Fig. 7a. Contest management.

– brief description,
– contest duration (for each division),
– contest start and stop times,
– team, proctor, division, file submission, and analysis mode parameters,
– age break for legal participants,
– contest type and time zone.

15. Current Status

The system has been in production for half a decade with complete functionality as described in this paper. The system currently manages 998 tasks (most from the past; a few hundred for the future) and has 96 users who can edit ‘probgate’ (the contest problems). Another half dozen instances support other organizations. An additional 61 users (some of them the same as the previous users) are working in ‘traingate’ to create more training materials for USACO. During March of 2009, 21 different coaches performed 3,343 different actions (task reading, task submission, etc.) in the system.

Contests are ready on-time, with requests for translation sent out 3–4 days before the competition begins. A typical season of six contests now yields – for the entire year – perhaps half a dozen queries for clarification (usually from new competitors) and, in general, no more than a single grading protest.



Edit contest MAR09

Prev. Contest Name	FEB09
Long contest name	Elite 2009 March Competition
Explanation	Contest for experienced programmers. Three divisions: Bronze (for almost everyone), Silver (by invitation), and Gold (by invitation).
GOLD Contest Duration (mins)	180
SILVER Contest Duration (mins)	180
BRONZE Contest Duration (mins)	180
Contest Start Time	18:00 Mar 12, 2009
START	Started 24 days 16:15:33 ago. [GMT: 0h00 3/13/2009]
Contest Stop Time	6:00 Mar 17, 2009
END	Ended 20 days 4:15:33 ago. [GMT: 12h00 3/17/2009]
Security Email	kolstad@ace.delos.com
Server Name	ace.delos.com
Max Points	10
Teams in this contest?	Yes [✓ No]
Proctored	Yes [✓ No]
Proctor DB Key	
Division invites required?	[✓ Yes] No
#FILE in this contest?	Yes [✓ No]
Instant Analysis Mode	Yes [✓ No]
This year's seniors grad yr	2009
Database Path	/home/kolstad/trainweb/ldb
Contest Type	IOI [✓ USACO] SPEED
Timezone	USA Mountain

Web Contact Rob Kolstad : Phone 877-753-3567
Copyright © 2005 Rob Kolstad

•

Fig. 7b. Contest configuration options.

16. Current Deficiencies

As systems do, the problem management system requires a facelift and internal update to bring it into the 21st century:

- Task timing (a function of the Linux kernel) is not reliable enough.
- The web display for contest tasks is too large; only filtering saves it from being too unwieldy to be used.
- Database navigation has efficiency, paradigm, and speed issues.

- The web displays are not as stylish or as functional as they need to be to be truly professional.
- The navigation among pages is inconsistent and poorly formatted.
- It is not yet possible to create one-off custom contests for trainees.
- It is not yet possible to create 'programming bees.'
- The older training pages are not integrated into this system and need to be.
- Logging is not always performed properly.
- The interface for 'confirmers' is not complete.
- The system should perform automatic tracking of 'solution times' for coaches who develop tasks.
- Processing the end of a contest still contains several manual steps; these should be integrated into the contest management subsystem.

17. Conclusion

While an IOI consumes six tasks (or perhaps 8–12, depending on one's point of view) in a year, the USACO consumes more than six dozen. The continuous development cycle has exposed many of the important factors in creating quality contests:

– data validation,
– multiple solvers,
– contest implementation without schedule pressure,
– automated contest configuration,
– running the USACO contest schedule would not be possible without this level of automation.

Our most important item for near-term future development is custom contests for training using old contest tasks.

The USACO system has proven successful for half a decade and complements the growing set of other systems available on the internet and contributes to the diversity of contests and environments that enable the field of competitive informatics to continue to grow and mature.

References

Diks, K., Kubica, M., Radoszewski, J., Stencel, K. (2008). A proposal for a task preparation process. *Olympiads in Informatics*, **2**, 64–74.

Verhoeff, T. (2000). *RobIn for IOI I/O*, unreleased draft, August.
`http://olympiads.win.tue.nl/ioi/twg/robin/Doc.txt`

Verhoeff, T. (2008). Programming task packages: peach exchange format. *Olympiads in Informatics*, **2**, 192–207.

R. Kolstad consults for the TAEUS corporation on intellectual property issues surrounding software and hardware. Rob is the head coach of the USA Computing Olympiad, and is also the head judge at the Pikes Peak Regional Science Fair. Rob earned his PhD in software from the University of Illinois at Urbana-Champaign after completing an MSEE at Notre Dame and undergraduate BASc degree from Southern Methodist University. Rob earned his Ph.D. in software from the University of Illinois at Urbana-Champaign after completing an MSEE at Notre Dame. His undergraduate BASc degree from Southern Methodist University was among the first computer science bachelor's degrees offered in the United States.