A new design of a system for contest management and grading in informatics competitions

Bojan Kostadinov[1], Mile Jovanov[2] and Emil Stankov[2]

[1] Faculty of Electrical Engineering and IT, Karpos II bb, 1000 Skopje
bojankostadinov@gmail.com
[2] Institute of Informatics, FNSM, Gazi Baba b.b., 1000 Skopje
{mile, emil}@ii.edu.mk

Abstract. Competitions in informatics are usually synonyms for algorithmic programming contests. Many of these competitions use automatic grading of the contestants' solutions. This is done by running them on batches of input data and testing the correctness of the output. In this paper we will introduce a newly developed system called "MENDO", which is used by the Macedonian Computer Society in the organization of national informatics competitions. We will outline the main modules every grading system should support (sandbox, grader, controller, auxiliary modules) and present how each one of them is implemented in our system. We will show that MENDO is comprehensive and superior in a number of functionalities as compared to the other presently existing systems.

Keywords: programming contests, algorithmic problems, grading systems, automatic grading, task evaluation, contest management system

1 Introduction

Competitions in informatics were introduced about forty years ago, with the idea of attracting talented young people to the science of computer programming. These competitions are usually synonyms for algorithmic programming contests (other types include architecture, design, development, specification, assembly, testing scenarios, etc). The closest thing to these algorithmic programming contests from the point of view of competition problems are the Olympiads in Mathematics.

Competitions in informatics have a long tradition in Macedonia. There were 20 national contest cycles till the end of 2009. After many competitions on national level, the best contestants represent themselves and Macedonia at IOI – International Olympiad in Informatics, BOI – Balkan Olympiad in Informatics (for high school students), and JBOI – Junior Balkan Olympiad in Informatics (for primary school students).

Usually, these programming competitions require students to submit programs which are then run through a variety of test scenarios and judged accordingly. The difficulty however, lays not so much in the programming but rather the design of the underlying algorithms [1]. A sample task can be seen in the appendix at the end of

this paper. More often than not, these contests are based on automatic grading of the submitted solutions. This is accomplished by running them on batches of input data and testing the correctness of the output. Time and space limits are usually enforced during the process, which allows to judge not only by the (approximation of) correctness of the solution, but also by its time and space complexity [2]. Besides the automatic grading capability, a contemporary programming contest system requires some other functionalities that would facilitate the preparation of the students for competitions, like some kind of a communication page (or a forum), a page for sharing appropriate learning materials, an info page, etc.

2 Contest management and grading systems

Historically, in Macedonia for example, a few separated systems (evaluation program, contest website, forum page called "Communication window", etc) supported the organization of the competitions. However, a lot has changed since the first competitions – the computer equipment has improved, new information technologies have been introduced, and at the same time, the technique and technology of development and judging competition tasks has improved.

At the International Olympiads in Informatics a use of a grading system is an absolute necessity. Let us reflect on IOI'1998 in Setubal, Portugal, with about 250 contestants, 3 tasks and 20 test cases average for a task. For each test case 2 executions have to be performed: execution of the solution with the corresponding test case input and execution of the checker (typing the output of the contestant on the screen and comparing it, visually, with the judge output is even more time consuming). That means a total amount of about 30000 executions. This is why the grading process in Setubal started at 3 P.M. and finished at 6 A.M. on the next day. The solution of the problem outlined above is obvious – use a software system for organizing and implementing the evaluation process [3]. The improvement is best seen in Fig. 1.

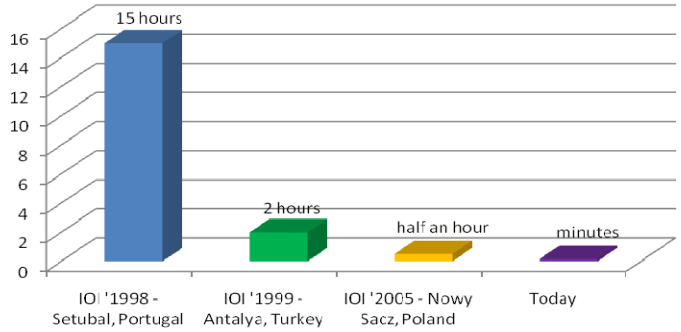

Fig. 1. Bar chart showing the time required for grading around 1000 algorithmic programming solutions on the IOI competitions (in chronological order)

The most suitable approach, from a practical point of view, certainly is to integrate all the required functionalities mentioned above in a single automated system for

complete contest management. This system would provide all the necessary contents for the students to successfully prepare and participate in the programming contests, and also all the necessary capabilities for the organization of programming contests (including submission of tasks and automatic grading).

3 Present contest management systems

There are few existing systems for contest management and grading. Traditionally, we still refer to them as grading systems. Today, however, we can talk about self managed systems that completely administer contests (including gathering and grading of submissions, managing competitor's questions and clarifications, publishing results, providing statistics, etc). Some systems even offer a continuous on-line training process.

Whatever the system offers, it should be designed such that it is simple, robust, secure and flexible. Standard modules for almost all grading systems are:

- **Sandbox** – ensures that the execution of a submission will not harm the system or the host computer; enforces time, memory and network restrictions. The sandbox is really the "heart" of every grading system.
- **Grader** – does compilation of contestant solutions, management of the sandbox, comparison of user output to correct output, and grading of a submission.
- **Controller** – handles the communication with the judges and competitions, and executes database operations.
- **Auxiliary modules** – handle printing, backups, storage, etc.

Table 1 summarizes some popular systems with their significant characteristics. As it is shown, American and Moe do not implement ACM style grading, while PC^2 does not implement IOI style grading. With the exception of PC^2, all the systems are dependent on the operating system that is used.

Table 1. Comparison between some of the popular grading systems

	pluggable graders	IOI style grading	ACM style grading	operating system	front end
American	Yes	Yes	No	Linux	web browser
Moe	Yes	Yes	No	Linux	command line
SIO	Yes	Yes	Yes	Windows	desktop app
PC^2	Yes	No	Yes	Win, Linux	java app

4 Architecture and analysis of the proposed system

Here, we describe the newly developed system called "MENDO", which is currently used by the Macedonian Computer Society in the organization of national informatics

competitions. MENDO was developed following the goal of integration of all previously used module functionalities in one compact environment. It was enriched with new features endorsed by the new technologies.

4.1 Architectural design

As shown in Fig. 2, MENDO includes all the modules that are standard for the contest management and grading systems. The implementation specifics for each of these MENDO's modules are outlined below:

- MENDO's **sandbox** uses P/Invoke (Platform Invoke) signatures and Win32 functions to create processes and group them in jobs. Jobs have the ability of limiting process privileges and resources.
  As a job object allows groups of processes to be managed as a unit, we use them to limit processes count, time and memory usage, security, etc. Job objects are namable, securable, sharable objects that control attributes of the processes associated with them. Operations performed on the job object affect all processes associated with the job object.
  The Linux variant of the sandbox is based on the Moe-Eval system, and uses Linux kernel calls (ptrace, ulimit, etc) to run the contestant programs in a controlled environment [4].

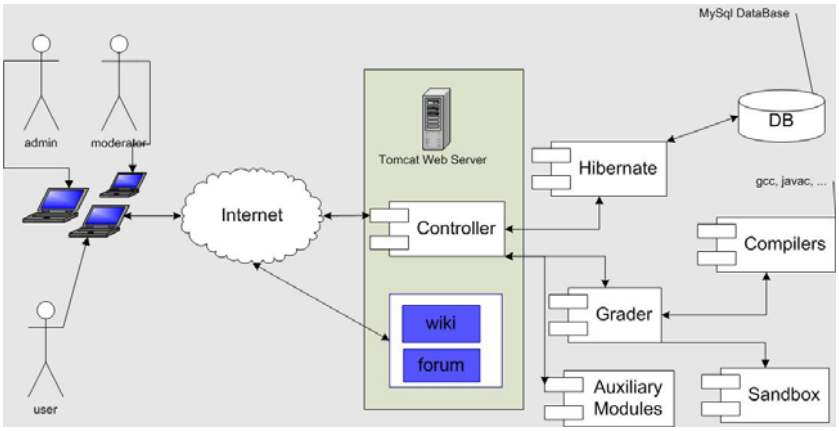

Fig. 2. MENDO's architectural design – the communication between the controller, sandbox, the graders and the auxiliary modules

- The **grader** is written in Java, and can be run on every popular operating system (like MS Windows and Linux)
- Judges and competitors use their web browsers to communicate with the system through a web application written in Java – the **controller.** For database operations we use Hibernate and C3P0 thread management to connect to a database and execute queries.

- In our implementation, the **auxiliary modules** are part of the web application and are also written in Java and run on an Apache Tomcat server. Besides the main system, MENDO also contains a public forum and a wiki.
  Every user is allowed to post content on the wiki, and to upload materials, images and other files. Only administrators and moderators are allowed to delete content. The same approach applies to the forum. As a wiki engine, we use JSPWiki, which is a feature-rich and extensible WikiWiki engine built around the standard J2EE components. JForum is used as a discussion board system (forum).

4.2 Specific MENDO features

An important feature of MENDO is that it uses cookies to support SSO – Single Sign On across multiple applications (the main system, the wiki, and the forum). All applications are running on the same server. Since all subsystems use Single Sign On and automatic language detection, there is no need to change the language or login every time you switch from one subsystem to the other, since this is automatically done by the underlying system.

MENDO uses 3 levels of caching (database caching – as provided by Hibernate, JCS – "Java Caching System" as secondary cache, and its own submission caching system).

Other MENDO's specific features, can be summarized as follows:

- MENDO is one of the rare systems that can operate on Microsoft Windows. Besides the fact that only a small number of grading systems operate on MS Windows, we found the OS to be very stable, reliable and easily controllable. The system can also be run on any popular Linux distribution;
- MENDO easily distributes load. Plugging more graders is easy, thanks to the modular architectural design of the system;
- It controls the entire system of the Computer Society of Macedonia, by providing automatic backups for itself and the other applications, self-tests of the application, the server and the operating system;
- The system has multilingual support (currently Macedonian and English, but we are planning to add more languages in the near future);
- MENDO is managed by several administrators and moderators, each with his own privileges and responsibilities. Every moderator and administrator can add tasks, create competitions, generate reports for each task and competition, and initiate system backups;
- There is heavy use of AJAX to simplify user interface operations (during registration, training sessions, competitions, etc). We use the jQuery 'write less - do more' javascript library to implement most of the event handling and AJAX operations;
- The system is designed and created by using free software (Java, Apache Commons, Hibernate, Struts, Java Caching System - JCS), and runs "entirely" on free software (Tomcat, MySQL, JForum, JSPWiki);

- MENDO was designed to be an entire gateway for algorithmic related topics, by including a news page, forum and an open wiki for publishing results, solutions and programming related materials.

4.3 Employment and performance of MENDO

In practice, MENDO is used as:

- a training system (contains tasks from past contests, both national and international);
- a contest management system (for organizing official national competitions and open online tournaments);
- Macedonian algorithmic programming gateway, containing a news page, a lot of programming related materials (organized in a wiki), and a public forum.

The training system ***can be used 24/7***. Every time a user logs in to our web site, he can view all the tasks that are available for training, and possibly submit a solution. After a solution has been submitted, the submission is added to a queue and judged as early as possible (this is ***no longer than 20 seconds***, even during heavy-load competitions). After a submission has been judged, the results of every test case are shown to the user. There is no limit to the number of submissions a user can make during a time period, but the system does ***support a couple of defensive mechanisms to prevent Denial of Service attacks***. A screenshot of the MENDO training system is shown in Fig. 3.

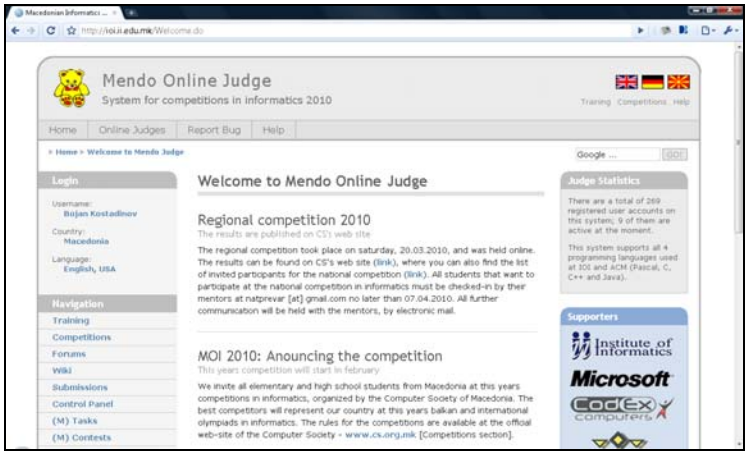

Fig. 3. Screenshot of the MENDO training system (made with Google Chrome). You can visit the system online at http://ioi.ii.edu.mk

As a contest management system, we have used MENDO in the 2010 contest cycle (more than 10 competitions, including both online and onsite rounds), and the system

proved to be quite stable, fast and robust. However, we plan to add a lot more features for the 2011 contest cycle.

Looking at the official rules of the International Olympiads in Informatics (IOI), we can conclude that MENDO ***supports every IOI rule and requirement***. It contains modules for detailed feedback, group testing, automatic competition management, statistics, backups, printing, and a lot more.

One other thing that is worth mentioning here is the ability of the system to produce reports and statistics. After the grading of each competition, the results are automatically published (this can be disabled, if necessary), and a report is generated. This report contains details and statistics for every competitor, every programming language, every task and even every test case that is part of that competition.

We tested the system in great detail, carried out many experiments and compared the system with the most popular grading systems. Table 2 is an extension of Table 1 and shows a comparison between MENDO and the grading systems that were previously analyzed.

Table 2. Comparison between MENDO and other popular grading systems

	pluggable graders	**IOI style grading**	**ACM style grading**	**operating system**	**front end**
American	Yes	Yes	No	Linux	web browser
Moe	Yes	Yes	No	Linux	command line
SIO	Yes	Yes	Yes	Windows	desktop app
PC^2	Yes	No	Yes	Win, Linux	java app
MENDO	Yes	Yes	Yes	Win, Linux	web browser

Using the open source software Apache JMeter, which is a 100% pure Java desktop application designed to load test functional behavior and measure performance, we have tested MENDO's 3 levels of caching architecture. Our architecture ***passed the test of over 5000 concurrent users***. The support of so many concurrent users is more than enough because one can rarely find a competition with so many participants. At the International Olympiads in Informatics there are usually around 300 competitors, and at ACM competitions even less (and they compete in teams, so in reality there is only one connected user per team).

We also made an experiment and ran about 2000 submissions through the system (we used a task given at our national competitions and the contestant's solutions to the task – executed enough times). The results we got from running the submissions on just 5 machines (at IOI, more than 30 machines are used per competition day) showed that the system can easily handle the load and that the time it took to test the solutions, generate the results and the statistics is comparable to all other grading systems outlined above. By supporting virtualization and/or implementing several graders on one machine, it is possible to lower the time required to test these solutions even more.

5 Future improvements of MENDO

In the near future we plan to address several remaining issues, including time measurement, reactive tasks, hacks, java security and large input/output. We also plan to completely rewrite the sandbox and add a new Rich Internet Application front-end to the system, based on Adobe Flex.

Time measurement is one of the most problematic issues that arise in grading algorithmic solutions, and is something that has been discussed in several papers. The current implementation of time measurement, based on the 'time' command, which can either return the real (or wall clock) time, or the user+sys (or instructions+system calls) time, can be improved by measuring the time of the solution programs several times and taking the minimum of those values.

Reactive tasks are a relatively new addition to programming contests. MENDO currently supports three types of tasks: reactive tasks, standard/batch tasks - or tasks for which the contestant needs to submit a source code of a program that reads the input data from a file or keyboard, and writes the results to a file or the monitor; and output-only tasks - or tasks for which the contestant already has the input data and just needs to submit the output.

Restricting java applications is another big issue, because the Sun JVM (Java Virtual Machine) performs a lot of operations that would normally be blocked by the kernel module [5]. The option of using virtualization to protect the machines on which we execute solution programs could be additionally considered. With the introduction of new multi-core processors, it can also be worthwhile to setup several graders on just one machine. This will greatly lower the time required to test solution programs, but serious testing has to be done before this can be used in a real contest environment.

A chat module is something that probably will greatly improve the system (a lot of popular systems, like TopCoder's Arena already allow communication). Implementing such a module can even ease the communication between contestants and organizers, and even between the organizers themselves.

Finally, a support for more competition modes can be added. Currently, only IOI and ACM grading styles are supported, but there are other popular grading options that can be used in real competitions (like TopCoder's grading based on submission speed). Also, another option that can be considered is allowing solutions in more programming languages, even though the system already supports all languages allowed at IOI and ACM competitions.

6 Conclusion

In this paper we discuss the need for contest management and grading systems. We have developed and presented a contest management and grading system called “MENDO”. We outlined the main modules every grading system should support (sandbox, grader, controller, auxiliary modules) and showed how every one of them is implemented in the MENDO system. This system is comprehensive and superior in a number of functionalities as compared to the other presently existing systems.

MENDO is currently used by the Macedonian Computer Society in the organization of national informatics competitions. Other contest organizers can smoothly adopt this system for use in their national Olympiads in informatics as it provides all the necessary functionalities for various forms of IOI style competitions. MENDO supports every IOI rule and requirement. Additionally, as it is web oriented, it can be used for organization of online world wide competitions.

Appendix: Sample task

Fence (ACM ICPC 2001. Northeastern European Region)

Workers are going to enclose a new working region with a fence. For their convenience the enclosed area has to be as large as possible. They have N rectangular blocks to build the fence.

The length of the i-th block is L_i meters. All blocks have the same height of 1 meter. The workers are not allowed to break blocks into parts. All blocks must be used to build the fence.

Input: The first line contains one integer N ($3 \leq N \leq 100$). The following N lines describe fence blocks. Each block is represented by its length in meters (integer number, $1 \leq L_i \leq 100$).

Output: Write one non-negative number S - maximal possible area of the working region (in square meters).

Sample Input	Sample Output
4	28.00
10	
5	
5	
4	

References

1. Burton, B.A.: Informatics Olympiads: Challenges in Programming and Algorithm Design. In Proc. Thirty-First Australasian Computer Science Conference (ACSC 2008), Wollongong, NSW, Australia. CRPIT, 74. Dobbie, G. and Mans, B., Eds. ACS. 9-13.
2. Mares M.: Perspectives on Grading Systems. Olympiads in Informatics 2007 Vol. 1. pp. 124-130 (2007)
3. Manev K., Sredkov M., Bogdanov T.: Grading Systems for competitions in programming. Thirty-eight spring conference of the Union of Bulgarian Mathematicians (2009)
4. Mares M.: Moe - Design of a Modular Grading System. Olympiads in Informatics 2009 Vol. 3. pp. 60-66 (2009)
5. Merry B.: Using a Linux Security Module for Contest Security. Olympiads in Informatics 2009 Vol. 3. pp. 67–73 (2009)