Olympiads in Informatics, 2008, Vol. 2, 122–130

Manual Grading in an Informatics Contest

Wolfgang POHL
Bundeswettbewerb Informatik
Ahrstr. 45, 53175 Bonn, Germany
e-mail: pohl@bwinf.de

Abstract. Bundeswettbewerb Informatik, the central informatics contest in Germany, from which German IOI participants are chosen, is not an olympiad in informatics (OI) in a strict sense. It has a wider range of tasks than OIs (including tasks without programs), and it uses a manual grading approach with grading schemes. Such a scheme is described for two example tasks, one of them an OI-style task, the other a data modeling task without programs and programming involved. Finally, some thoughts are added on how manual grading and tasks without programs could be applied to IOI.
Key words: informatics contest, grading, manual grading.

1. Black-Box Testing in IOI Competitions

In the annual IOI competitions, tasks are of an algorithmic nature. They require the participant to write a program that is to constitute a solution of the task. For each task, the source code of the corresponding program is submitted. The quality of a submission is determined by checking submitted programs against a defined set of test data. Each test case is determined by input data and output data. A test case is satisfied by a submission if the submitted program outputs the test output data when applied to the test input data. Hence, if a submission achieves full score, it can be said to reproduce the input-output relation given by the test data – no more, no less. The contestant, who aimed at solving the given problem, cannot count on a full score to confirm the solution to be perfect.

There has been criticism of black-box testing in general and the IOI grading approach in particular; see, for instance, Cormack (2006), Forisek (2006), Verhoeff (2006). It can surely be said that quality and choice of test data influence scores dramatically. There may be a flaw in submissions that will remain undetected because there are no suitable test cases, and positive properties of submissions may not be rewarded. In addition, the black-box testing approach regularly leads to severe punishment of a submission that implements correct ideas but shows a slight implementation mistake.

Recently approaches were suggested and tried to improve that situation. For instance, test case bundles were introduced, where each bundle should be designed to cover a specific desired property of solutions. Thus, test case design ought to become more focused on qualitative assessment of a submission, in contrast to the more quantitative focus of mere efficiency testing with test cases of different size.

2. Manual Grading in Bundeswettbewerb Informatik

In Bundeswettbewerb Informatik (short: BWINF, Engl.: Federal Contest in Computer Science), a manual grading process is used, which focuses on qualitative assessment of submissions. This contest was described in (Pohl, 2007); by using some of the dimensions for characterising contests suggested by Pohl (2006), it can be summarised as a task-based contest with homework rounds, mixed submissions of texts and programs (source code plus executable), and manual grading.

In the first two rounds of that contest, a submission consists of the following parts:

1. A required part of any submission to a task is a written description of the solution approach.
2. The majority of the tasks also requires the submission of a program; in this case the written part should also explain how the solution approach was implemented, typically by short descriptions of the most important program components.
3. Contestants are demanded to demonstrate the functionality of their submission with examples. Often, task formulations contain a set of required examples, but contestants are always asked to invent and demonstrate their own examples or test cases.
4. This is complemented by printouts of source code, which may be inspected by judges if the other parts of the submission leave doubt about how to grade the submission, or if they want to understand the reason for a mistake, etc.

Grading of BWINF submissions is done on a grading weekend, when all jury members meet. The group event allows the jury members to immediately clarify possible open issues with the jury chairman. Submissions are graded individually and sequentially by two jury members. The grading itself is organised as follows: For each task, a set of grading criteria (a grading scheme) is developed by the task committee. This set may be refined after looking into selected submissions; real submissions may often contain unexpected flaws or, rarely, unexpected solution approaches.

BWINF uses a “negative grading” scheme: Grading criteria are formulated to discover errors and weaknesses of submissions. In the first round, for instance, judges start with a score of 5 points for each task. For each grading criterion that is met, 1 or sometimes 2 points are subtracted. The overall score must not be negative; hence, task scores range from 0 to 5 points. Participants will be informed about grading results with respect to the criteria: They receive an individual score sheet that lists all grading criteria and states which of these were met by their submission and, hence, led to score deductions. In addition, they receive a text that explains solution approaches and the meaning of the grading criteria.

3. Example Tasks

In this section, two example tasks will be described, each from a first round of BWINF. The first could have been an olympiad task as well, it presents a typical algorithmic problem. The second task is of a completely different style; it does not require a program, but asks for information and process models.

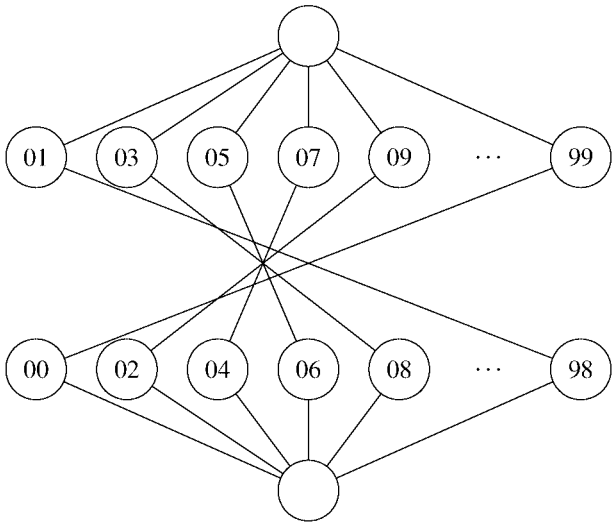

Fig. 1. A partial view of the constant size flow graph (102 nodes, 350 edges), with nodes for cent values only.

3.1. *Example 1: Prämienjagd (Bonus Hunt)*

3.1.1. *Problem*

This task was given in the first round of the 26th BWINF 2007/2008. It could have been an olympiad task as well, since it presents a typical algorithmic problem. The task can be summarised as follows:

- Imagine yourself in a supermarket, at the cashier. You have to put your goods on the belt, one after the other. The supermarket will grant you a bonus for every neighbouring pair of goods the prices of which sum up to an amount with a decimal part of 11, 33, 55, 77, or 99 cent. Each good can contribute to only one pair. Given a list of prices, write a program that computes an order of the prices so that the number of bonuses generated by matching pairs is maximal. It should output the pairs.
- *Example*: For the price list [1.99, 4.13, 6.64, 8.98, 9.91, 1.99], two matching pairs can be found: (4.13, 8.98) and (6.64, 9.91).
- Test your program with the input data given on the BWINF home page and with your own meaningful test data.

3.1.2. *Solution Approaches*

There are many solution approaches that find the best order and hence a maximum set of pairs. But for larger inputs, only few solutions worked. The given test data contained up to 500,000 entries. However, it was stated on the home-page that, in order to achieve full score on this task, it was not necessary to submit a perfect solution which would work for even the biggest cases.

First, it can be observed that a price with an even cent value can only be combined with a price with an odd cent value – and vice versa. So one solution would be to compute a maximum matching in the bipartite graph of prices with odd and even cent values, with edges between prices that can be paired. But the crucial observation is that only cent values need to be considered. Then, a flow graph can be constructed with nodes for all cent values, edges between possible pairing values (like 33 and 44, since they add up to 77), and source and target nodes, with edges between source and even values, and between odd values and target, respectively (see Fig. 1). The number of prices with some specific cent value determines the capacity of the edge between the price node and the target or source node. Capacities of edges between price nodes can be infinitely high. To such a graph, a network flow algorithm can be applied to compute the maximum number of pairs, and this computation takes a time that is independent from the size of input data. In the end, prices can be paired according to the computed flow.

Interestingly, the problem could be solved with a greedy algorithm, too. Also in the case of applying a greedy algorithm, constant run time can be achieved by considering cent values only.

3.1.3. *Grading Scheme*

For grading this task, judges were given a grading scheme with a list of criteria and the corresponding score malus (or bonus, in one case). They are listed in Table 1, with explanations given in italics if necessary. The table also states, how many percent of the 292 submissions to this task met each individual criterion. It is interesting to see that, also in this case of manual grading, test cases play a crucial role. Both the response to the given test cases and self-invented test cases were found to be lacking in about half of submissions. Furthermore, a statement about performance was missing in 37 % of submissions. It looks like many contestants did not find it necessary to investigate into the complexity of their solution, neither by theoretical argument nor by running their solution on given or self-invented test cases – or had the idea that such an investigation might discover the suboptimalities or errors of their solution.

3.2. *Example 2: Supermarkt* (*Supermarket*)

3.2.1. *Problem*

This task was given in the first round of the 25th BWINF. Its story refers to a supermarket setting, too – but be assured that, in general, BWINF task settings are taken from a wide range. The task can be summarised as follows:

> In a food market, at the cash-point price labels are scanned. Non-packed goods like fruit and vegetables will be put on the scales, and the clerk will manually input a product number. The cashier system should cover the following business cases:
>
> - output a receipt;
> - output a list of goods where the amount in store is lower than a given threshold;

Table 1
Grading Scheme for BWINF-task “Prämienjagd”

Criterion	Malus	Applied to
submission does not discover that pairings can be limited *not all possible combinations of prices need to be considered; odd-cent-prices can be paired with even-cent-prices only, and better: a price can be paired with another one only if their sum has one of the legal cent values.*	−1	19%
program too inefficient for other reasons *Even if the approach does not consider too many price combinations, it may still use to much memory, or fail to solve larger cases for other reasons.*	−1	19%
submission does not make a statement about performance *Since it was obvious that performance was a crucial issue in this task, participants should make at least a rough assessment of the (time) performance of their solution.*	−1	37%
incorrect results *The submitted program is expected to produce correct results. This basic criterion and its score malus should only be applied when incorrect results could not be attributed to other criteria.*	−2	31%
program does not output the pairs themselves	−1	2%
submission does not contain outputs to the published test cases	−1	46%
self-invented test cases are missing or are not meaningful	−1	51%
problem reduced to looking at cent amounts only *A very extraordinary case in a BWINF grading: Submissions that discovered and presented the crucial idea that led to an optimal performance were rewarded with a score bonus.*	+1	30%

- output the top-seller list of the month, with goods sorted by product groups;
- output labels with addresses of those clients who used their bonus card and bought a significant amount of wine recently. The labels shall be used for a promotional letter.

What data does the system use and how should it be organised? How can (on that basis) the output jobs be done? What do you think about the last business case?

That is, this task did not ask for a program. A submission was required to describe a data model as well as processes based on that data model. In addition, it is a general BWINF requirement that each submission should present a number of examples sufficient to illustrate and explain their ideas.

3.2.2. *Solution Approaches*

The starting point for a data model is the product number. In the task formulation, it is only mentioned for non-packed goods, but also packed goods should have such a number. This number is the key to all further information about the products of the supermarket. Then, the sub-tasks can be solved as follows:

receipt For each product number, we need to know a name and a price. The price is per pack or per kg (for non-packed goods). For non-packed goods, the price on the receipt is computed from the price per kg and the weighing result.

shortage list For each product number, we need to know the current amount and the minimum amount in store. To make the list informative, we would also like to know whether the product is packed or non-packed; then the shortage list may contain "pack" or "kg", resp. The information needed to update the current amount can be obtained directly from the cashier.

top-seller list This business case requires product groups, so that product numbers need to be related to product groups. Furthermore, every day for each product number the amount sold on that day is stored. That is sufficient to compute the sold amount per month (and might help in cases where you would like to produce top-seller lists per week-day etc.). From this information and the relationship between product number and product group, the top-seller list can be generated.

promotion addresses In this case, individual client data are needed. Similar to the product number, a client number is introduced. For clients with bonus card, address data are known and related to the client number. Moreover, for each client and each product, we store how much the client bought of that product, e.g. since the last promotional letter concerning that product (there may be smarter solutions, but this works for our case). For every product group, there is a minimum amount a client should have bought in order to receive a promotional letter.

An assessment of the last business case should not only consider economical aspects, but also take a critical position concerning privacy aspects. Of course, promotional letters must only be sent upon clients' consent.

3.2.3. *Grading Criteria*

The grading scheme for this task with its list of criteria is given in Table 2. Again, the table also states, how many percent of the 218 submissions to this task met each individual criterion. Interestingly, the more formal aspects of the task (the data model and its description) appeared to be less problematic than the assessment of the business case or the (very basic) difference between packed and non-packed goods. The last grading criterion helped to detect the lacking interest or awareness of the contestants concerning privacy issues.

Table 2
Grading Scheme for BWINF-task “Supermarkt”

Criterion	Malus	Applied to
no difference between packed and non-packed goods *The data model does not make a difference between packed and non-packed goods. That is a mistake, since non-packed goods are treated differently in many cases.*	−1	35%
lacking data model description *The model should be described using some (semi-)formal notation. The description must identify the key data, how other data can be accessed using key values, and how separate sets of information (products, product groups, clients) are linked to each other. Depending on how severely a submission misses these requirements, judges may subtract 1 or 2 points.*	−1 / −2	21%
lacking data model *The data model does not allow to produce (some of the) outputs that were specified in the business cases. Depending on how severely a submission misses this requirement, judges may subtract 1 or 2 points.*	−1 / −2	28%
no description of output procedures *Not only product and client data, but also procedures operating on that data are needed to produce the required outputs. Those procedures must be explicitly described, too.*	−1	6%
lacking assessment of last business case *The assessment of the last business is not acceptable if it does not consider privacy aspects or does not require clients' consent.*	−1	39%

4. New Ideas for IOI?

4.1. *Manual Grading*

BWINF experience shows that manual grading can be applied successfully in an informatics contest. The requirement to create a list of grading criteria (whether negative, positive, or both) forces task committees to make their reasoning about the problem and about the quality of a submission explicit. Such a grading scheme also allows for feedback to the participants. And it can be applied to a wide range of tasks, for which black-box testing is impossible.

However, the BWINF approach cannot easily be applied at IOI, because it heavily relies on the participants explaining their solutions in natural language, and on jury members being able to read and understand that solution. But without a suitable mechanism for (partial) translation of submissions (at least of source code), black-box testing remains the only choice. Manual grading is possible in international contests, like the example of the International Mathematics Olympiad shows. In an international contest, a grading process involves translation work of delegation leaders or even grading work of delegation leaders and is prone to be biased by their interference. For IOI, it would be important to design a manual grading process which would avoid such bias. For instance, delegation

leaders could be asked to translate only, not to grade. The grading then could be done on the basis of the translation, of-course double-blind, by leaders of other delegations.

Very interesting suggestions along this line were made by Verhoeff (2006). Verhoeff's central proposal is to introduce a "thoroughly prepared and motivated grading scheme, supported by measurements". This would follow the example of BWINF, and I fully support Verhoeff's argument.

4.2. *Tasks without Programs*

Here, we are speaking of tasks that would not involve any computer program. This is different from tasks without programming; e.g. a task that would require contestants to test and detect the flaws of a given program with their own test cases, would be a task without programming, but not a task without programs.

For tasks without programs, the formats used in answers to the task's problem(s) strongly determines how the task can be handled and submissions can be evaluated. If the set of possible answers is clearly defined, or the format of a correct answer can be automatically detected, automatic grading is possible. In all cases where the answer set cannot be clearly defined (like with natural language answers), manual grading is required, and all above-mentioned problems of manual grading apply.

Furthermore, in tasks without programming, the internationally understood "linguae francae" of Informatics, the programming languages, disappear. For tasks that involve proofs concerning properties of more or less mathematical constructs (like graphs), the usual mathematical notations could be used. For tasks that involve data models (like our example above), UML or ER-diagrams might help – but should IOI require knowledge of such notations from its contestants?

My personal opinion is that tasks without programs are possible within IOI. However, the BWINF task above is not a perfect example. In particular, its grading criteria are fairly vague and offer jury members too much freedom in grading.

References

Cormack, G. (2006). Random factors in IOI 2005 test case scoring. *Informatics in Education*, **5**(1), 5–14.

Forišek, M. (2006). On the suitability of programming tasks for automated evaluation. *Informatics in Education*, **5**(1), 63–75.

Pohl, W. (2006). Computer science contests for secondary school students: Approaches to classification. *Informatics in Education*, **5**(1), 125–132.

Pohl, W. (2007). Computer science contests in Germany. *Olympiads in Informatics*, **1**, 141–148.

Verhoeff, T. (2006). The IOI is (not) a science olympiad. *Informatics in Education*, **5**(1), 147–159.

W. Pohl was educated in Computer Science, and received a PhD in 1997 from the University of Essen, Germany. For many years, he investigated the use of Artificial Intelligence techniques for the improvement of interaction between humans and machines. In 1999, he changed position and perspective by becoming Executive Director of the German Federal Contest in Computer Science. Among his responsibilities is to coach the German IOI team and lead the German IOI delegation. Now, his interest lies in improving Computer Science contests, establishing new ones, and work on diverse other projects, everything in order to popularise Computer Science among youth. Hence, he co-ordinates the German participation in the international contest “Bebras”. From 2003 to 2006, he was elected member of the IOI International Committee, and briefly held the position of Executive Director of IOI in 2006.