



Utrecht University – Faculty of Science

Master's Thesis

Finding winning patterns in ICPC data

A retrospective analysis of programming competition results

Author:

Rick de Boer

r.h.deboer@students.uu.nl

1st Supervisor:

Dr. C. P. de Campos

c.decampos@uu.nl

2nd Supervisor:

Dr. M.J.S. Brinkhuis

m.j.s.brinkhuis@uu.nl

*This thesis is submitted in fulfillment of the requirements
for the degree of Master of Science*

in

Business Informatics

July 23, 2019

Abstract

The International Collegiate Programming Contest is a worldwide, multi-tier competition where students from universities all over the globe compete to be the best programmers. These competitions result in data in the form of scoreboards, which are never analyzed and collected before. The purpose of this master thesis is to gather and structure all this data, and provide insights. It details characteristics of highly performing teams, compares different competitions and looks deeper at the way problem characteristics influence each other. This thesis answers the question how competitions compare and what patterns exist throughout the recent years.

Results show that competitions can be explored and summarized using the measures of popularity and difficulty and a visualization technique to show the distribution of solutions over a single year. This information and other metrics are used to compare within and between regions, where the European competitions are found to be most similar to the World Finals. Further analysis was done on the best performing teams, where was found that they are better in all aspects of their game compared to other teams; they on average require less attempts, solve faster, can handle more difficult problems, are slightly more efficient, and hand in more solutions. Finally, several different ways of gathering information problem descriptions were used, where was found that it is difficult to build a model that is both accurate in identifying algorithmic topics and has a high recall. A model that with reasonably high precision was made and run on some ICPC problems, where was found that (of the most prevalent topics) problems labeled as being about *data structures* are most significantly different from others. All these results can be used by multiple stakeholders of the ICPC.

Document outline & File storage

This thesis is structured as follows. First, a research plan is detailed, where the scope, problem statement, research goal, research questions and research method are described. After this, a literature and a context study are conducted. Then, the process of data collection and cleaning is described, where the ETL process is detailed. After this, data analysis is performed, and results are given to answer all research questions. Finally, a conclusion, discussion and possibilities for future work are described.

The data used in this research is stored at <https://github.com/RickdeBoer/ICPC-Scoreboards>, and other generated information and complimentary files such as the visualizations for each year and the submission platform are stored at <https://github.com/RickdeBoer/ICPC-Thesis>.

Foreword

At the start of this project, I had no notion of the existence of the International Collegiate Programming Contest and only vaguely knew about competitive programming in general. I have always had an interest in programming and data analysis, so when I was searching for an interesting and suitable thesis subject, I looked for a project in this direction. Then I met Cassio, who introduced me to the world of competitive programming and the unexplored and scattered dataset of the ICPC, where eight months ago, I became enthusiastic on researching this competition. Now, at the end of this project, I can finally present all efforts of collecting and researching the competition data in the form of a master thesis, and an accompanying paper about the data set collected and cleaned will also be published in the near future. It has been quite a journey and I am proud of this achievement. I would like to take the opportunity to thank some people that have been involved in this journey with me.

First and foremost, I would like to thank Dr. Cassio de Campos, my first supervisor, without whom I would not have been able to successfully complete this project. He has kindly devoted his time, was always reachable and involved, and through meetings and feedback, helped improve the quality and analysis of this research. His expertise on the context and data of the competition was of great value, and his involvement in this project is highly appreciated. I would also like to thank Dr. Matthieu Brinkhuis for being my second supervisor and being involved in this project.

I would like to thank my friends, who have supported me on this journey by listening to my stories and having discussions on the content and research opportunities. Finally, I would like to specially thank my family and parents, for motivating me and always being there when I needed them. Their involvement and support meant the world to me.

With the support of all these special people and with extensive effort, I have managed to research and write about the biggest project in my life thus far, captured in the thesis laying in front of you; a retrospective analysis on the ICPC.

Ad majorem Dei gloriam.

- Rick de Boer, July 2019

Contents

Abstract	i
Document outline & File storage.....	i
Foreword	ii
List of acronyms	v
Chapter 1: Introduction	1
Chapter 2: Research plan	3
2.1 Problem statement.....	3
2.2 Research goals.....	4
2.3 Research questions	5
2.4 Research method	6
2.5 Planning.....	7
2.6 Literature research protocol.....	7
Chapter 3: Literature and context study	9
3.1 Literature study	9
3.1.1 Strategies.....	9
3.1.2 Winning characteristics and practical tips	11
3.1.3 Programming contest statistics analysis.....	12
3.2 Context information	13
3.2.1 International Collegiate Programming Contest	13
Chapter 4: Data description	15
4.1 Data processing.....	15
4.1.1 Notes on data peculiarities.....	16
4.2 Data structure.....	17
4.2.1 Data structure description.....	18
4.2.2 Data content description.....	19
Chapter 5: Descriptive and comparative analysis	26
5.1 Concepts and general patterns	26
5.1.1 Comparative measures	26
5.1.2 General patterns in solution distributions	27
5.2 Competition descriptions	28
5.2.1 World Finals	28

CONTENTS

5.2.2 Europe	30
5.2.3 Latin America.....	33
5.2.4 South Pacific	35
5.2.5 North America	37
5.3 Competition comparison.....	41
5.3.1 Problem set comparison	42
5.3.2 Country comparison	43
5.4 Performance of top teams.....	45
5.4.1 Origin of the top 10 teams	45
5.4.2 Performance of the top 10 teams	46
Chapter 6: Expanding data and analysis	51
6.1 Basic meta-information.....	51
6.2 Topic detection.....	53
6.2.1 Gathering expert input	53
6.2.2 Topic extraction with machine learning.....	55
6.3 Analysis with generated data	66
Chapter 7: Conclusion	68
Chapter 8: Discussion	71
Limitations.....	71
Research notes.....	71
Future work.....	72
References	74
Appendix A: Data sources	77
Appendix A.2: Sources for problem pdfs.....	81
Appendix B: Patterns in problems	82
Appendix C: Solution distribution graphs.....	84
Appendix D: Model scores	85
Acknowledgements.....	88

List of acronyms

In order of appearance in the text:

ICPC: International Collegiate Programming Contest

KDP: Knowledge Discovery Process

CRISP-DM: Cross-Industry Standard Process for Data Mining

IOI: Olympiads in Informatics

ACM: Association for Computing Machinery

UVa Online Judge: An online automated judge hosted by the University of Valladolid

ETL: Extracting, Transforming and Loading

CSV: Comma Separated Value

NA: North America

CERC: Central European Regional Contest

NEERC: Northeastern European Regional Contest

NWERC: Northwestern European Regional Contest

SEERC: Southeastern European Regional Contest

SWERC: Southwestern European Regional Contest

ERD: Entity Relationship Diagram

LDA: Latent Dirichlet Allocation

API: Application Programming Interface

TF-IDF: Term Frequency-Inverse Document Frequency

MNB: Multinomial Bayes

SVM: Support Vector Machine

K-NN: K-Nearest Neighbor

CNN: Convolutional Neural Network

NNET: Neural Network

Chapter 1: Introduction

According to Forbes¹, there has never been a better time than now to be a programmer. With an increasing amount of jobs as software developer being available, it is expected that there will be 27.7 million developers worldwide by 2023², an increase of little over 20%. People studying computer science or other related studies should fill this upcoming gap in the future. But programming is more than just a job. Besides the growing demand for programmers in the field of engineering, programming is also an activity which could be done purely for fun and/or as a hobby. As it turns out, the popularity of recreational programming amongst students is also increasing. Especially for computer science students, participating in programming contests is likely the most rapidly growing extracurricular activity (Rivella, Manzoor & Liu, 2008).

One of those popular competitions is the International Collegiate Programming Competition (ICPC), an annual, multi-tier competition held amongst college students on a global scale, with world championships finals organized every year. Last year alone, almost fifty thousand students from three thousand universities participated in ICPC regional competitions (The ICPC Foundation, 2018). This number has steadily increased over the past years. Because it is a competition of significant size with a lot of talent and skillful people involved, big companies are interested in the competition and the competitors. For example, IBM has been the main sponsor for the past ten years (now only for some competitions), and Huawei is the current main sponsor of the European competitions. It should therefore be no surprise that, according to ICPC Executive Director Bill Poucher (Poucher, 2017), a great number of ICPC alumni end up working at important positions such as CTO, CEO, or President at major companies. He adds examples of remarkable ex-competitors, stating that the first engineer at Google was an ICPC world champion and the first CTO of Facebook came second at the world finals. This illustrates that, besides the fun and competitive aspect, the competition also has a practical use, where it can be used as a steppingstone for a further career in IT. In essence, this means that the competition is a platform that functions as an intermediary to connect potential employer and potential employee. At one hand, it provides an opportunity for students to excel and show their skills. At the other hand, companies can recruit from this pool of talent to find programmers who have proved their capabilities.

But, to get high results and catch the attention of those companies, teams must prepare. Besides practicing coding in general, part of this preparation is gaining knowledge about the competition, which will be helpful in order to find out what they can expect. Doing so by looking at results, tactics and patterns of previous competitors would be a logical approach. However, as it turns out, this information is not readily available, as there is little or no scientific work written down about past competition years. This leaves an interesting research opportunity, as all involved parties can benefit from this information. Furthermore, collecting,

¹ <https://www.forbes.com/sites/quora/2017/01/20/will-the-demand-for-developers-continue-to-increase/#7d62ebe133ee>

² <https://www.daxx.com/article/software-developer-statistics-2017-programmers>

cleaning and storing this information provides opportunities for the community to do additional research. As part of this analysis, there will be looked at the possibilities to generalize the results to the learning of algorithms and programming, by analyzing the relations of the competition problems and computer science topics. This might lead to insights for improving education about programming and better understanding of what topic need to be focused on to be prepared as well.

In conclusion, it is useful for all stakeholders to gain knowledge and statistics about past competition results. Besides creating knowledge about past competitions, it is interesting for the organization of the competition as well, as it provides them with information on how to present themselves towards their participants and sponsors. But mostly, teams could prepare themselves better and in turn could achieve higher results, making them more skilled and also interesting, such that companies would hire potentially more capable employees.

Chapter 2: Research plan

This section describes the problem setting, research goals and research questions which will be answered in the later chapters. It also discusses the literature research protocol that will be used during the literature research. This will justify this research, show what exactly will be researched, how that will be done and define it all as a plan.

2.1 Problem statement

The main artefact that is a direct result of the programming competition is the scoreboard. The result of a competition and all team's programming submissions are summarized in there, where the final stance, score for each team and information about the way each problem is solved is kept. As stated by Bloomfield & Sotomayor (2016), competition scoreboards can be analyzed to discover patterns and analyze the performance of teams. However, the collection and cleaning of this data seems to be a problem in itself. Throughout the years, competition data from regional and global ICPC competitions has been collected and stored spread across the internet. The ICPC Foundation keeps track of only the minimal (but enough for score keeping) information of these scoreboards, such as the name, university and final scores of all teams. The full version of these boards for most regional competitions is however stored on their own websites. Data contained within these scoreboards is not always properly structured yet, and the notation and format of it has not been consistent between years and regions, making analysis at the current time impossible. As known to the author, no attempts have been made so far to process and research these scattered sources of data.

There does exist some literature that uses these ICPC scoreboards in some way or another. One (of the few) example is from Manzoor (2006), who uses the minimal world final scoreboards of 1998 till 2005 to compare country performance. The absence of literature and research using this data is however a strong indication that there is unexplored territory to conquer. Looking even further by studying teams and their elaborated scores could lead to discovering more detailed patterns and thus deeper understanding of the data and the competition.

In addition, information about team tactics, winner strategies and retrospective analysis are also almost not available or kept. For example, any strategy followed appears to be based purely on what works best for a given team, in contrast to using known approaches. This statement is confirmed by Amraii:

"Each year, they use different methods and strategies, based on observations of previous year's strengths and weaknesses, to develop better teams for the next year's competition. These techniques are completely experimental and are not based on conventional methods."
(Amraii, 2007, p. 1)

Strategies used seem to be short sighted and derived from internal improvement and possibly from theory, rather than other techniques. They can be thought of up front, but it remains unclear if and how they are indeed used in practice. Furthermore, if teams wanted to consult past results or best practices of other teams, they would have needed to go to a lot of effort as this information is not readily available. But, there does exist other practical work to aid the

contestants, such as the book by Halim, & Halim (2013) about the theory behind the problems used, a guide about common mistakes made during competitions by Manzoor (2008) and some work on strategies for handling problems at a theoretic level by Trotman & Handley (2008), who were the first to propose the best strategy to take in terms of problem solving in different scenarios. These strategies are very interesting to look at, because they are one of the only other factors to influence besides theoretical knowledge and teamwork. There is also some other work on this topic, for example by Bloomfield & Sotomayor (2016) and Ernst, Moelands & Pieterse (1996), who's papers both give advice on possible team structures and strategies to follow. However, as it turns out, there exists no scientific literature and information about best practices based on analysis of previous competition data.

This research is performed to fill this gap of unidentified patterns and aims at creating a deeper understanding about the competition to assist the teams competing and the organization itself. The problem and task at hand can be summarized by the problem statement formulated according to the design science template from Wieringa (2014):

How to collect, clean and analyze competition data to satisfy the need for information about past competitions so that teams competing can perform better and there is more information available for all stakeholders involved in the context of the International Collegiate Programming Competition?

The two main problems and tasks become apparent here. First, data is not yet kept in a central repository and therefore needs to be collected and structured. Second, the competition is not using their previous scoreboards to their full potential yet, so an analysis of this data could provide them with new insights.

2.2 Research goals

The aim of this thesis is threefold, as depicted in Figure 1. At one hand, the ICPC data is gathered, cleaned and prepared for analysis, structuring it in a format ready for research now and in the future. At the other hand, the data of different competitions is described, and everything is analyzed to find patterns. As it will be the first look into this data, this research has an explorative nature. Patterns this thesis will specifically focus on are finding characteristics and strategies of winning teams, comparing different competitions and analyzing the problems posed during those competitions. This information is aimed at supporting the ICPC- and research community with retrospective patterns not seen and used before, and helpful knowledge for winning the competition.

In order to reach those goals, an overview of existing literature and recent information about programming competitions, strategies and competitions tips will be given. Together with an analysis of the competition data, a complete and comprehensive overview of practical and helpful knowledge will be provided, giving a retrospective on past competitions.



Figure 1: Research goals.

2.3 Research questions

Derived from the problem statement, the main research question is formulated as follows:

***MRQ:** What winning strategies and characteristics can be identified in the ICPC data in order to enhance the programming competition community with knowledge about patterns existing throughout the years?*

This research will specifically focus on analyzing the years 2012 up till 2018, because those years are the most recent, are all available, and should provide up to date knowledge on recent patterns. Expanding beyond this timeframe to include earlier years is not significantly beneficial or doable for several reasons. First, as the competition progresses and changes as the years progress, earlier information is less relevant for today's competition. For example, the difficulty of programming contests seems to increase (Forišek, 2010). In addition, the ICPC is divided into several regions. Not every region has properly kept record of their data, and especially from earlier years a lot of information is missing. Because of the data availability and the fact that the author has the most affiliation with- and knowledge about the western competitions, this research will focus at the more western regions; the Latin-American, European, South Pacific and North American regional competitions and the World finals. The latter is more a competition with no bound physical region but is considered as a special region in this context.

In order to answer the main research question, the analysis is categorized into several topics that more or less built upon each other. These topics have been formulated into a set of research questions (RQs) stated and explained below.

***RQ1:** What are the general patterns in the previous years?*

The first analysis task is to describe the data and to find general patterns. This means characterizing competitions by their statistics and looking at describing and summarizing meta-information about competitions, problems and teams' performance. This in itself is a question, as a way to summarize and describe a competition needs to be found. Looking out for other local patterns that could potentially be found within the data is another objective here. Combined, this provides the basic explorative retrospective of the competitions for the most recent years.

***RQ2:** What are characteristics of winning teams?*

Then, the characteristics of outstanding teams will be evaluated. Finding distinctive trades of teams that are performing relatively good will lead to more understanding of the most interesting placing groups, thereby showing others what aspects could positively impact their performance. Also, these teams are significantly different from the others by their performance, so most can be learned from them.

***RQ3:** Are there differences in regional, or regional and World Final data?*

Found characteristics and patterns will subsequently be compared by looking at different competition regions and the world finals to identify general trends or differences. This comparative analysis will show how regional competitions are associated in terms of e.g. difficulty, popularity and rivalry, but also evaluates and relates it against the most important

competition of all, the world championship. This will result in more insights into what competitions are more (dis-)similar, and how they relate to the world finals.

RQ4: What insights can be gained from the problem descriptions?

Finally, all data gathered up till this point will be expanded with additional information about the problems that are solved by the teams during the competitions. This information will be tried to be derived from problem descriptions using natural language processing techniques. It should provide deeper insights on the meta-information from each problem, focused on algorithmic topics the problems are about. This can in turn be related to problem difficulty, winning teams and other patterns for additional insights.

2.4 Research method

In order to fulfill the research goal of creating publicly analyzable data, any used method must be reusable, comprehensive and complete. This way, data will be prepared and made available in a reproducible manner, providing possibilities for further research. It is therefore important that the research methodology that will be used, is known and used in practice, thus being proven to be useful. For this project, the task at hand is one of discovering knowledge in a database, formally known as a Knowledge Discovery Process (KDP), so data mining methods must be considered and evaluated.

According to Niaksu (2015), the Cross-Industry Standard Process for Data Mining (CRISP-DM), SEMMA process model, and Predictive Model Markup Language (PMML) are the most prominent domain independent KDP methodologies. Of all methodologies, CRISP-DM is considered to be one of the most popular (Azevedo and Santos, 2008; Shafique & Qaiser, 2014; Onwubolu, 2009). In practice, this data mining process model is also considered to be the most widely used, as confirmed by a KDNuggets poll where 43% of the respondents (the largest share) voted it being their mainly used methodology (Piatetsky, 2014). Providing data mining analysts with a complete toolset (Niaksu 2015), this method seems to be suitable for this project, as it is well known, often used and complete, and fits the context; the CRISP-DM will therefore be used as research method for this project.

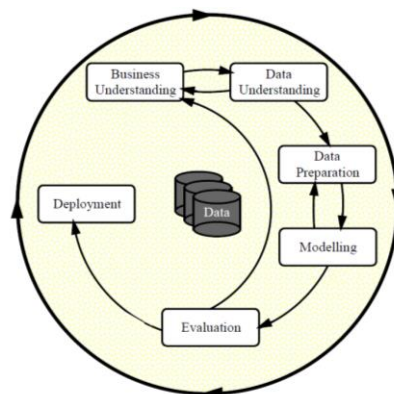


Figure 2: CRISP-DM, Wirth & Hipp (2000).

This method consists of several steps illustrated in Figure 2, starting with first understanding the ‘business’ context. This will be done by a literature study and looking at known information about the competition, such as articles, related research and the general rules of

the competition. This will lead to a better understanding of the data. Then, the data will be described, prepared and structured into a tabular “data analysis ready” format, leading to a good and well-structured set and database. While doing so, it will also help improve the context understanding (it is a cycle in the CRISP-DM process) as the data structure and contents will become more apparent. Using this data to build ‘models’, in this case structuring the data in different ways to be able to get insights, initial answering of the research questions will be possible, which is depicted in the methodology as the ‘modelling’ phase. In this context, it means building query models, thereby “asking the right questions”. After evaluating the results and answers, it should become apparent what further possibilities and opportunities exist for analysis and/or what implications hinder complete answering of the questions. This information will be used during the evaluation step to improve and expand the results of the first phase, and also to improve the context understanding. Finally, after a second round of analysis, an in-depth answer to all research questions can be given and conclusions can be drawn.

2.5 Planning

The high-level timeline for this project is marked by two distinct phases; the set-up of roughly three months and research phase of five months. After setting up, where the context is explored by a literature study and data is collected, cleaned and prepared, the real work starts in phase two. First, some additional data about the competition problems is gathered and a final version of the RQ’s and analysis questions is made. In order to have everything ready for the next phase, the data, tools and paperwork are finished here as well. Then, a paper about the data will be written and published to ensure that other researchers can also use and find the data, after which the data is explored to answer all research questions one by one. There will be accounted for the possibility to expand on the questions posed and answered during the research based on the information found in the data; there will be focused on what is most interesting to explore. At the end, following the CRISP-DM method, the results are evaluated and the improved. Finally, all the work done for this project will be written down in this thesis.

2.6 Literature research protocol

In order to get a good overview of existing research and provide some groundwork for the research questions, a literature review must be performed. In order to do this structurally and make the research more reproducible, a literature research protocol must be defined. The content for this protocol can be derived from the research questions, which are stated in a logical order that can be followed when researching literature.

2.6.1 Objective

As the literature review is the first step in understanding and analyzing the context, it provides a basis for the rest of the project. Furthermore, giving an overview of known information about the competition in the shape of a review of different kinds of literature, is an integral part of the research. These different kinds of sources include scientific studies and grey literature such as fact sheets and instructional books as well as the website of the ICPC foundation. Together, all information should result in a concise and practical overview of known best practices and

competition information, such that it is usable and readable for both people who know and do not know but are interested in the field.

2.6.2 Search strategy

When looking at the different topics for which literature must be found, both forward snowballing, backward snowballing and ad hoc ways of searching are used, where whenever additional topics pop up, literature about them is searched. This snowballing is a method described by Wohlin (2014), which is using the reference list of-, or citations to- a specific paper to further identify other relevant papers in the domain.

First, for the main research question, some general and contextual information about the programming competition must be given. This information leads to understanding of the research context and implications thereof. This same information can be used as a starting point for the first question *RQ1*, to get an understanding of the meaning of the data. For question *RQ2*, literature or articles describing how teams have won must be found. Part of winning is using a good strategy. As stated by Trotman & Handley (2008), "Of two equal teams, the team that chooses the better strategy will win". Arefin (2006) also states that a crucial component for a good team is defining and using a teamworking strategy. Strategies found in practice or strategies proven to be helpful will thus also be described. If no such information exists, then information about tips and tricks which helped teams in the past will be the starting point for further research. All information from *RQ1* and *RQ2* will then be combined as input for *RQ3*. During a preliminary literature study however, it was found that for *RQ3*, regional and worldwide were never compared before. Any other relevant information on research of one of both, so only regional or worldwide data, will therefore be groundwork for this question. Research on similar data will also be looked at, to see what techniques or results could be relevant for this project. Lastly, for *RQ4*, research about the problems used in the previous or other competitions will be looked at. Literature for all *RQ*'s will be explored in Chapter 3.

As with all research, finding work about similar work to this thesis could provide additional insights into ways to approach the research and could even result in additional data research questions. As seen in the CRISP-DM cycle, there is an explicit step to evaluate the work so far. This entails that, if at the end of the literature review (or analysis phase) opportunities for additional research questions arise, they will be considered and possibly added.

Chapter 3: Literature and context study

This chapter will explore literature on the topics described in the literature research protocol. Information found will be used as input for answering the research questions in later chapters. After this, a more in-depth look at the ICPC context will be taken, which is used as a setting explanation and starting point of the data understanding.

3.1 Literature study

There are several journals and conferences relevant for this research, of which some are worth pointing out. For example, although the ICPC does not have its own journal, the similar competition Olympiads in Informatics (IOI) does. Some research that considers the ICPC has been published there, and although most papers seem to be focused on learning and the educational value of programming competitions, other relevant studies can be found. Other main sources are the multiple journals of the Association for Computing Machinery (ACM), which has been the endorser of the competition (Constantinescu, Nicoara, Vladioiu & Moise, 2017) for many years (the full name of the competition is ACM ICPC). These sources will be used as a starting point for investigation.

3.1.1 Strategies

A starting topic for this research is looking at work on strategies. These could be evaluated with real-life ways (visible in the data) how people approached the competition and ultimately combined with other patterns found, resulting in a complete advice on a working approach on how to tackle the problems faced. Some relevant and interesting papers and their results are detailed below.

Ernst, Moelands & Pieterse (1996) base their insights on their own experience as contestants and observations in the field and looked in-depth into team setups and strategies. They provide helpful tips, with one of their main points being that one should focus on using the synergy within each team. This means harnessing the expertise of each team member, by letting everybody do what they can do best. Another point is that, as time management plays a crucial role, it is important to determine what problems you can solve while filling as much of the available time as possible. This also means that each problem must be completely solved in the same time, because even a 99.9% finished solution earns no points. Easy problems should be solved quickly, and large problems should be started immediately. The authors propose three example strategies: *the simple strategy*, where the key idea is working as much individually as possible, *terminal man*, where only one designated team member does the programming and the *think tank*, where two people analyze all problems up front (and more thoroughly later) and the third person does the programming. The latter strategy should be favored over the others, as it should help an equal team solve the maximal number of problems. Also, the letting a single person write a program should be the most efficient way. This statement is also confirmed by Mansoor (2001) and Chavey, Monrey, Brackly & Werth (as cited in Trotman & Handley, 2008).

Bloomfield & Sotomayor (2016) explain team dynamics and what to do during each phase of the contest. They identify four phases, the *opening*, *early game*, *mid game* and *end game*. During the *opening*, the easiest problems (at least one) should be identified. Then, in the *early game*, all these problems should be attempted and solved if possible, while also roughly ordering the remaining problems. The *mid game* is characterized by taking the longest, as medium difficulty and unsolved easy problems should be tackled. Finally, in the *end game*, all unsolved problems should be attempted once more and if there is time, the hard problems could be tried. The authors note that the length of each phase is strongly depended on the difficulty of a contest. Besides these phases, Bloomfield & Sotomayor also explain a strategy where two members work and program the current problem at hand and one person works and prepares the next. Pair programming should be beneficial for less experienced teams, while more skilled teams can choose when or when not do program together. A critical point made is that team members should be able handle constructive criticism of each other when working, as the authors had problems with that in the past.

Trotman & Handley (2008) look at strategies from a theoretical point of view, where all teams and team members are assumed equal, but also assume that problems are fixed to team members and cannot be transferred, and the computer is always available. While the first assumptions are reasonably valid, the latter two are not. Nevertheless, having these assumptions therefore only provides a theoretical perspective, where they are able to prove some interesting aspects. First, they show that not all problems may be solved if handled in order of increasing difficulty and therefore the investigation of possible strategies is valid. They find that a team should first determine how many problems they and other teams will likely solve. Through simulations, it is shown that when a team expects to outperform all others based on number of problems alone, reverse order of ease should result in winning. But, should there be a tie on the number of problems solved by multiple teams, then order of ease is most effective. If possible, problems should be solved in parallel as well. These advices are all highly theoretical and their effectiveness is also based on how well a team can estimate its own skills and the difficulty of each problem in a set.

Other work by Amraii (2007) details observations on teamworking strategies. Amraii groups these strategies into two distinct categories, *individual-based* and *group-based* strategies. The individual strategy is a more utopic strategy where every member picks problems and solves them correctly on their own first-try, but as stated, this is not very realistic. The group-based methods are more interesting. Amraii mentions three: the *specialization*, *duty-based* and *manager-based* method. In the *specialization* method, the person most proficient on a certain topic will solve that particular problem, with the benefit of having everybody work on their most comfortable problems and disadvantage of minimal teamwork. The *duty-based* method is different in the way that not whole problems, but tasks are divided by assigning roles (e.g. by having an algorist, coder and debugger). With the benefit of engagement by every team member on all problems, it is more difficult to decide what to do when solutions do not work or when deciding on what problem should be tackled next. Finally, the *manager-based* way of working focuses on producing code without errors and is similar to the *duty-based* method, with the addition of a manager. This person keeps track of all tried and solved problems and

makes decisions. Although this method should result in working code faster, it requires a lot of practice to get used to collaborating in this way.

Looking deeper at strategies, they can be described as a multivariate timeseries forecasting problem (Chatfield, 2000), where based on previous submissions, the current position compared to others and next optimal steps must be determined. It is multivariate, as other time variables (explanatory values such as problem difficulty and other team's problem-solving times) will influence the next step to take. Looking at ranking and predicting in more detail leads to better understanding a team's strategy at a lower level, but is not directly relevant for the current research. About ranking and predicting, there is some related work. It turns out the multiple different models and techniques are used. For instance, Maiatin, Mavrin, Parfenov, Pavlova & Zubok (2015) made estimations for the winners of the IOI depending on their passing score during the qualifications. Using linear regression, they find that the solvability index is a better approach than using expert opinions for correct estimation. Predicting any used strategy or the rank of teams however is another topic for future work.

As a concluding remark for this section, it could be said that there are multiple possible ways a team could strategize. The order of handling problems, task division and role assignment are differentiating factors that influences how well a team can and will perform. For now, it remains unclear how influential these factors exactly are and if this strategy theory is used in practice. The mentioned research however does indicate some promising ways a team could organize themselves, and considerations for choosing their strategy, that likely are beneficial.

3.1.2 Winning characteristics and practical tips

As winning teams have never been explicitly analyzed in retrospective, only advice in the form of tips and tricks can be found in the literature that can be used to further improve the chance of winning. The most helpful and non-obvious general tips (e.g. an obvious tip that is always stressed is to practice a lot, on problems of different algorithmic topics) found in the literature will therefore be summarized and listed. Giving an overview of tips from these authors results in the concise checklist given below. Teams could use this information for determining their strategy and improving their performance in general. There is not much more information available on this topic, further justifying the research at hand.

In his book, Arefin (2006) opens with some practical tips. Focus on learning only one programming language very thoroughly should avoid dirty debugging (meaning long debugging times). Furthermore, beginners should not focus on efficiency and try to solve a problem in as many ways as possible, instead of starting on other problems. A general rule flowing from this approach is to take and implement the simplest algorithm found. Also, all team members should be proficient in common algorithms and should be able to code them into correctly working programs. Another good point is to design test cases yourself instead of submitting them, which will make for less wasted submissions. Finally, looking at what problems other contestants are solving might give insights into the (perceived) difficulty of those problems, which could hint to what problems could be easier and therefore solved faster.

Manzoor (2008) states that every team should know the strengths and weaknesses of each member. To be effective, each member should be an expert on different topics. Ideally, two members should be adept on the same subject (Amraii, 2007) so they can work together. Another point made is that a big thing to avoid is real-time debugging. This is so-called the ultimate sin (Ernst, Moelands & Pieterse (1996), likely because it takes a lot of time and is error prone. As Burton (2007) puts it, any given solution to a problem must be correct and efficient as well, considering the running time on the computer to not exceed the time limit.

Bloomfield & Sotomayor (2016) give more practical tips for training. When practicing, it is good to have a setting as realistic as possible. They stress the use of real problem sets during five-hour sessions with multiple teams, where the hardware setup of the competition is used. This should mimic some of the unique factors of the competition. Also, teams should have problem discussion sessions and review official solutions to get better acquainted with the problems, and to understand all of them at the end of a training session.

3.1.3 Programming contest statistics analysis

Some analysis of programming (competition) submission data has been performed by other authors. Looking at their work will give ideas of how to approach the task at hand and possibly provide ideas for measures to use during the analysis.

Manzoor (2006) uses the statistics and submissions of UVa Online Judge till 2005 and also looks briefly at results (scoreboards) of the IOI and ICPC to gain information on programming contests and experience of contestants, with the goal of finding points of improvement for these programming competitions. In his paper, he answers several questions leading to interesting insights, the most similar to the research described in this thesis. First, he shows that with practice, the acceptance rate of submissions increases, compile errors decrease, and the percentage of wrong answers remain relatively stable, but these changes depend on the difficulty of problems. He also looks at the highest rank and average performance of countries that entered in the ICPC, where it turns out that in the last eight years, only teams originating from forty unique countries qualified for the world finals. Then, most frequent error sequences are given, where the most important results are that the same error types occur often multiple times (i.e. the same mistakes tend to be made in sequence) and when four incorrect answers are given, the subsequent submission is four times more likely to be incorrect than accepted. Finally, it turns out that in ten or less submissions, roughly 97% of the solutions are found and for contests of five hours, average completion time for additionally solved problems decreases.

Revilla, Manzoor & Liu (2008) look at the frequent response sequences submitted to a programming contest training website and try to design a programming competition by taking the best parts of ICPC and IOI, as well as online judging and training websites. Their research is partly similar to Manzoor (2006) and draws some of the same conclusions but differs in data used. A point they make is that of all submissions 36.73% of errors is informed and 18.14% uninformed, meaning that roughly one of three errors is not helpful for the contestants. More interesting is that number of compilation errors made decreases (and acceptance rate increases) with practice (the training website keeps track of the number of problems solved), but the rate at which these numbers improve is different for easy and hard problems. The main

point is that onwards from about fifty easy problems, it is better to practice on more challenging problems.

3.2 Context information

The competition being focused on during this research is the ICPC. According to Manzoor (2006) and the ICPC themselves, it is on the most popular and prestigious programming competitions to date, along with the International Olympiad in Informatics (and TopCoder, which is not taken further into account because of its very different and much smaller setup). Although these two competitions are both short-term and focus on problem solving, they differ in rules and setting, which should be taken into account. The most important differences are on team origin, educational background, team size, and problem setting. For instance, the IOI is a national competition where every country sends a single team of four, whereas the ICPC is a multi-tier, university-based competition for teams of three. Also, the IOI scores partial solutions and poses only three problems, while the ICPC only accepts or rejects solutions and poses usually eight to twelve problems. These points should at least be taken into mind when making statements about competitions in general. For now, the ICPC should be described in more detail to get a better idea of the data origin and problem setting.

3.2.1 International Collegiate Programming Contest

With roots tracing back to 1970, the International Collegiate Programming Contest (ICPC) is an annual programming competition for universities all around the globe (“About ICPC”, n.d.). It is a multi-level competition where teams of three students aim to outperform others in Regional Contests, after which they get to advance to the ICPC World Finals. The highest scoring team out of roughly 130 teams will become the world champion. The main goal and mission of the ICPC is to provide college students the opportunity to enhance and show their collaborative, programming and analytical skills (“World Finals Rules”, 2018). In 2017, a little under fifty thousand students from 3,098 faculties in computing disciplines, coming from 111 countries participated in ICPC Regional competitions. This number of participants is continuously growing and has increased by more than 2000% in the past twenty years (The ICPC Foundation, 2018).

Programming languages used in the world finals are Java, C, C++, Kotlin and Python (“World Finals Rules”, 2018), where C++ is the most dominantly used language. Regional competitions allow at least the same- and sometimes additional languages. Besides programming skills, knowledge of subjects such as basic algorithms, dynamic programming, simple computational geometry and basic combinatorial knowledge (Manne, 2000) is presumed and necessary to solve the posed problems.

Competition setting

During the contests, the contestants will be generally faced with a set of between eight and thirteen problems. In the timespan of five hours, as many problems as possible must be solved, where each solved problem is worth a point. Teams have complete freedom in choosing the order of the problems they want to solve. The problems are non-trivial and designed with the philosophy that every problem is at least solved once, that every team is able to solve a

minimum of one problem and no team is able to solve all problems (“About the Contest Problems”, 2018). In order to do so, problems are designed and selected of varying difficulty, where each problem is described (in detail) in a real-world scenario. The goal is to decipher the meaning of each scenario in order to find the underlying problem and design an algorithm to solve it. Along with a problem description, a representative input set and accepted output set are given (“About the Contest Problems”, 2018).

An interesting complication is that each team only will be provided with a single computer and calculator, so they will have to program in turns while also designing algorithms on paper. Besides some additional provided paper, pens and possibly an unannotated natural language printed dictionary, teams are normally not allowed to bring or use any other materials.

Once a solution to a problem is found, it can be submitted to be judged; such a submission is called a *run* (“World Finals Rules”, 2018). A run can either be accepted and marked as correct or rejected and marked with an error code. According to the most recent rules, there are three distinct marks for incorrect runs; *run-time error*, *time-limit exceeded* or *wrong answer* (“World Finals Rules”, 2018). These respectively mean ‘there is a problem with your code’, ‘your code is running too long’ or ‘your code gives incorrect output’ and give the contestants an indication of what to change in their submission. These error marks have changed over the years, for instance they no longer include *presentation error* (meaning ‘answer is in the wrong format’). An important note here is that every rejected run is given a twenty-minute time penalty, so it is key to not hand in too many wrong solutions. However, in practice, the amount of problems solved is of much higher importance than the amount of trial and errors.

General announcements of accepted solutions for each team are made during the competition. There is an online scoreboard where teams and their accepted runs are shown, but physical balloons in distinct colors for each problem are also handed out when teams solve a specific problem. At the end, the team with the most points and the least total time spend wins.

Chapter 4: Data description

The first research goal of this thesis is to collect, clean and store ICPC scoreboards scattered across the internet. This chapter will detail the ICPC data used in this project and the steps taken to process it into a format ready for analysis.

4.1 Data processing

As stated in the introduction, the focus of this research lies on gathering data from the European, Latin American, South Pacific and North American regions as defined by the ICPC, from 2012 up till 2018. These regions include several competitions, each being a bit different from the others. Data was collected from publicly available sources in the form of scoreboard tables. These sources are all listed in Appendix A. The high level ETL (Extraction, Transform and Load) process of the data is shown in Figure 3 below. Special (semi-)manual processing steps taken for each of those competitions and peculiarities are described in section 4.1.1, the general processing steps are described here.

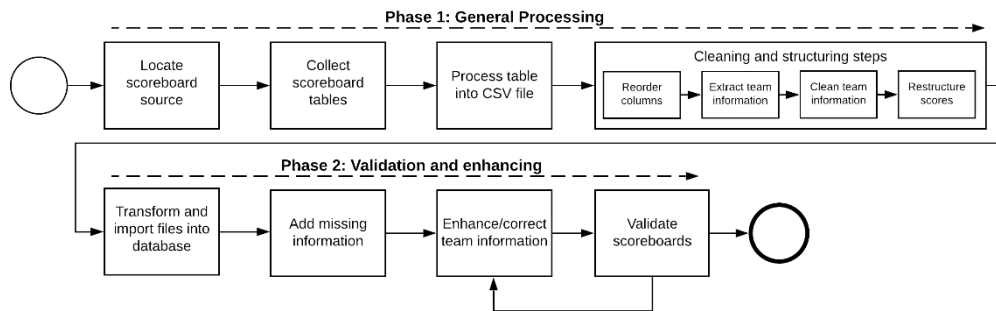


Figure 3: Data processing steps applied to each online source.

As is illustrated in Figure 3, the process starts with locating the scoreboard sources. This is a significant task in itself, as each competition has its own website and information is often not structured the same way. For older and missing data, web archives are consulted. Once the correct webpage is found, the data is downloaded in the format it is provided in, either a HTML or PDF page. These pages are then subsequently processed into CSV files, either manually or with the help of tools³. There was explicitly chosen to turn the collected tables into CSV files, because these are text based, easily readable by both human and computer and an often-used format for databases. Turning them into text made it possible to go over all teams, row by row, and alter each column separately based on some predefined rules. This significantly sped up the cleaning process.

Then, as the file structure was set, the next significant step of transforming the scoreboard files could begin. This most important and biggest step consists of multiple smaller steps, which are being carried out by special purpose written code. Cleaning was done by going over all rows one by one with a script and carrying out several actions, based on the given data and its structure. Note here that, although often almost all steps needed to be carried out, not every step was necessary for each single data set entry (it depended on the quality and format of the

³ Such as <https://www.becsv.com/table-csv.php>

available data). For instance, molding the columns into the same order was not always necessary, and the 'clean team information' step can consist out of multiple sub steps (such as adding, removing and restructuring data), but not all those steps were always carried out. Another significant step here is restructuring the scores. At least nineteen different ways of writing the number of solutions and attempts per problem have been found, and all of those are restructured into the same attempts/time format (e.g. when a team did 3 attempts and solved a problem in 35 minutes, their score becomes 3/35. This also means time penalties are ignored in the individual solution entries).

After a cleaned file for a certain year and competition was ready, the now roughly cleaned files could be transformed and loaded into the database, marking the start of phase two. The missing information about teams was filled by using external sources, which mostly consisted of adding the country from which the teams originated. Other examples of (semi-)manual taken actions are adding team names from the general ICPC website, removing competition site information, extracting university names out of an image or logo, and changing the country code from two to three characters.

The final and most time-consuming task was to correct and validate all information. These iterative steps were carried out several times, because oftentimes, new incorrections or impurities were found. The great majority of information about universities was cleaned extensively, very often manually, because university names were non-trivially abbreviated in the data and not always written in the same way (e.g. 'University of Utrecht' and 'Utrecht University'). The goal here was to have all universities written in a uniform way. This iterative way of enhancing and validating was repeated until the data was found to be correct and complete. Finally, the now properly cleaned database was ready for analysis.

4.1.1 Notes on data peculiarities

This section details the noteworthy and special characteristics of some of the data, which must be kept in mind when interpreting results. The general cleaning process as described in Figure 3 was also followed for these regions. Regions and years not listed here can be assumed to be fully available in the correct format, such as the World Final data, for which no special cleaning steps were necessary.

- For Northwestern Europe, companies that participated in certain years were excluded.
- For the South Pacific region, the scoreboards from the first division were taken (there are two divisions in this region), because those winners advance to the World Finals (the ICPC has both divisions integrated into a single scoreboard).
 - For other competitions that had multiple divisions, this rule was also applied.
- The North American region consists out of thirteen competitions, but only eleven of those are included in the dataset:
 - The North American Qualifier contest was excluded because this competition has different rules than all the others. For example, a team could consist out of minimal one and maximal ten members, whereas normally this would be always three.

- The other excluded competition, the Northeast North American Regional Contest, has no publicly available data and could therefore not be included.
- For the Mid-Central North American Regional Contest from 2012 up till 2014, no full versions of the scoreboards can be found online. These years were therefore chosen to not be included, so no data is available for these years.
- Although (extensive) manual efforts have been made to properly clean the data in a uniform way, there are possible differences with the online ICPC data:
 - Because of different ways of writing and encoding, there are minor differences in the names of teams, e.g. a space is sometime a -, and accents or diacritical marks are sometimes omitted.
 - Following the convention of the ICPC, the language in which the university name of each team is written is kept. As the original sources were used, the university names therefore are also often in the language of the source, e.g. a Brazilian university such as the 'University of Brasilia' is often written in Portuguese as 'Universidade de Brasília', which is also the case in the official ICPC results. Although different ways of writing the same university in the same language are filtered out, there might be some overlap in the same university being mentioned in different languages (e.g. 'Universiteit Utrecht' and 'University of Utrecht' refer to the same university but are counted as different universities).

4.2 Data structure

After all separate files (one or each year and for each competition) were imported into a database and extensive cleaning was finished, all data was finally ready. To assure that information was not stored more than necessary, the competition meta-information was stored separately from the competition results. This has two advantages. First, there is minimal repetition of the fields (e.g. all competition data does not need to be mentioned for each team row) and second, the data has a similar format to that of the ICPC foundation. This resulted in the data structure as shown in the ERD of Figure 4.

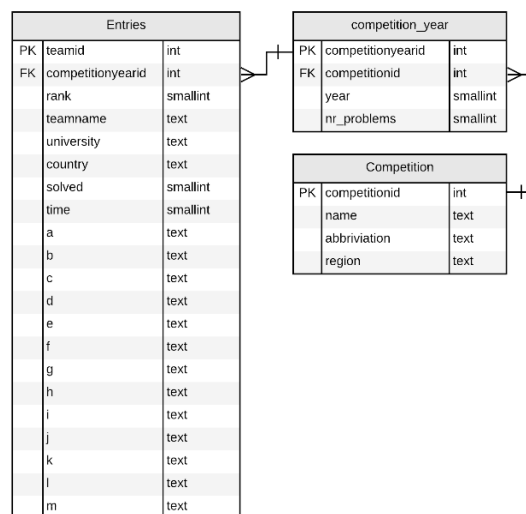


Figure 4: Data structure after import.

Here, the *Entries* table contains team information, each team’s final score and separate scores for each problem. The *Competition* table contains some info about each competition, and the *competition_year* table functions as an associative table to connect both.

However, the current format has the disadvantage that problem information cannot simply be queried. Therefore, in order to make the data more easily analyzable, it must be restructured. The *entries* table containing all information about- and from- each team will therefore be split into two, where the scores from each team and problem are kept separately from the information about each team. This creates some additional foreign key fields, but separates everything by subject, creating a structure with the least overhead which is best extendable and queryable.

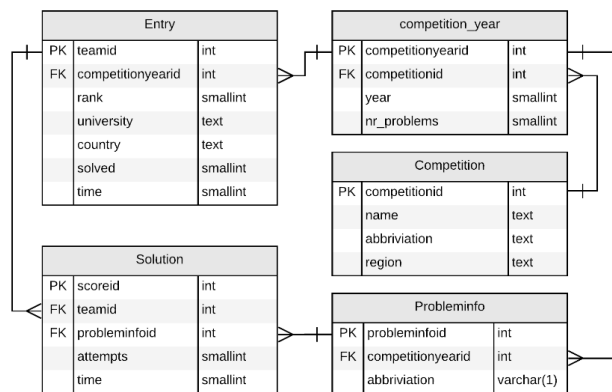


Figure 5: Final data structure.

4.2.1 Data structure description

As shown in Figure 5, the data is divided by topic for easy extraction and/or combination of desired information, but also for optimal storage in the sense of least repetition of information. In this context, an *Entry* represents a team and all its information, which has entered in a single competition. This also includes the team’s final rank for that competition, their final score, consisting of the number of problems solved, and total time taken. This is the time elapsed from the beginning of a contest till the first accepted submission of a problem, accumulated for each problem, including a twenty-minute penalty for every additional attempt on each solved problem. Note that, in contrast to the team information normally present in the public ICPC data, team names in the original sources are not always the same as the official ICPC data, so they are included in this dataset but not further used for analysis. Moreover, a team’s name is no essential information for the analysis, since it is not directly associated with any other information (the same name can even be used by different teams and/or different team members). Looking further at the data structure, a *Competition* contains metainformation such as its region, the years it was held (i.e. the years that are present in the data) and the number of problems that each year’s problem set have had. Next, a *Solution* represents all input from a single team for a single problem, where the *attempts* are the amount of times a team tried to solve a problem and the *time* is the total time it took to solve. If no time but only the number of attempts is present, a team did not solve a specific problem. Finally, the *Probleminfo* table is separated from the other information, also with RQ4 kept in mind, as more information will possibly be added there.

4.2.2 Data content description

Before looking deeper into the data, an influential statistic must be shown. Figure 6 shows the relative proportion of contestants that solved at least one problem compared to those that did not solve anything. These teams could significantly impact the later analysis as they have a big influence on averages (e.g. the average amount of solves), and this amount fluctuates, as can be seen in the figure. Most striking here are the biggest and smallest numbers, where the East-Central NA- and Mid-Atlantic USA- Regional Contest show a relatively high number of teams of almost a third that were not able to solve any problems. In contrast, the South Pacific region only has a one-fiftieth part of the teams without a correct solution. But this region does also have a much lower number of teams participating, which could be the reason that the average contestant is more motivated and/or skilled.

The reason why there are competitions where the proportion of the teams without any solutions is large could be due to several reasons, e.g. it could be that teams enrolled and never showed up or attempted some problems but were not able to solve anything. That the proportion of such teams differs is likely because competitions include them differently; some might have left them out and some kept them in the data, but these large differences could also reflect the type of problem sets that are used in different regions or the general skill level of teams. Of the 2083 teams that did not solve anything present in the dataset, 590 did not make any attempts. Whatever the exact reason is, this number has impact on a fair comparison of performance per competition, so these teams are left out of the general analysis for now. Only teams that look like they take the competition seriously (that not just show up to make numbers or for the free food) and show that they at least were active by making one or more attempts are considered of interest; they are defined as *active* teams. This also ensures that least amount of data is excluded from the analysis. From now on, when referring to teams, the group of *active* teams are meant.

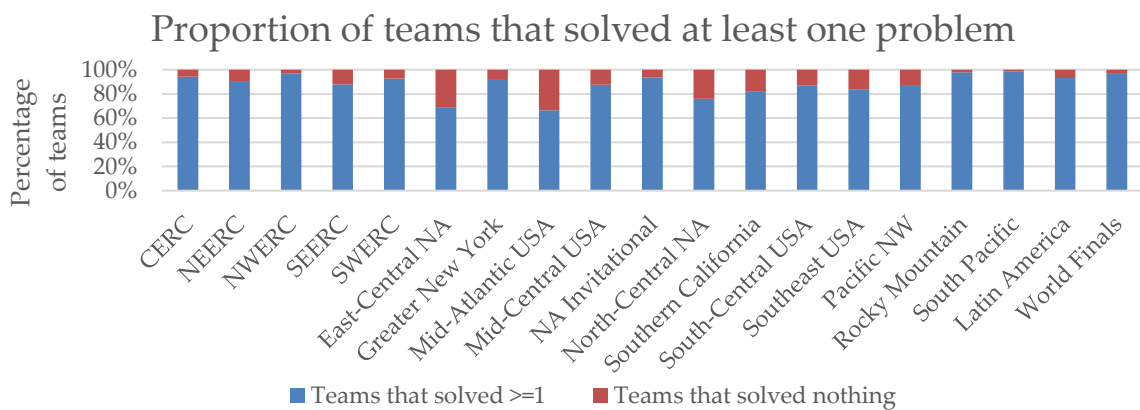


Figure 6: Teams that solved at least one problem compared to teams that solved none.

In total, the data consists of 14554 team rows (15141 total minus 590 excluded inactive teams) which provided 60544 solution rows. These teams participated in 129 unique matches, from 23 different competitions (aggregated into 19 in this dataset) of 5 distinct regions; Europe, Latin-America, North America, South Pacific and World Finals. Those matches had 1362 problems in total, which are 10.573 problems on average. Note that not all competitions from

the North American region are part of this dataset, because some of them were not publicly available. Also, the years 2012 to 2014 for the Mid-Atlantic USA Regional Contest and 2014 from the South-East USA Regional contest are missing because of unavailability.

4.2.2.1 Competitions and regions

Region	Competitions	Total teams	Active teams	Average active teams per competition
<i>Europe</i>	5	3950	3846	110
<i>Latin America</i>	1 ⁴	2735	2714	78
<i>North America</i>	11	7345	6884	92
<i>South Pacific</i>	1	233	233	33
<i>World Finals</i>	1	877	877	125

Table 1: Teams and competitions per region in the dataset.

To give an overview of the distribution of participants over the regions and to indicate the popularity and scale of the competition, Table 1 shows the total number of participating teams for each region. This information is at regional level; some regions may have many more participants in prior/qualifying levels, but these are not present in this dataset; only regional finals are considered. To be able to compare them, an average is given, where can be seen that on average, the World Finals is the biggest region in terms of absolute number of teams entered, followed by Europe and North America. Table 2 further details the participant’s background, showing the most frequent countries (out of the 89 total) where the teams originate from. These and all other participating countries are illustrated in Figure 7.

Country	Nr. of Teams	Country	Nr. of Teams
USA	6590	ARG	238
RUS	999	KAZ	207
MEX	483	POL	202
BRA	470	AUS	194
CAN	458	CUB	192
BOL	298	DEU	192
COL	292	GBR	183
UKR	289	FRA	168
PER	260	ROU	160
CHI	241	NLD	144

Table 2: Top 20 most frequent countries.



Figure 7: Origin of the participating teams.

A team’s country of origin is often directly related to competition region and its university, as teams normally compete in their local competitions. It should not be surprising that the biggest countries on earth also happen to be the most frequent in the dataset. This is likely because there are more competitions (also more universities and people) in the USA and this dataset;

⁴ All five regional competitions of the Latin American region are aggregated, as these competitions are held at the same time and use the same set of problems. This essentially makes it a single big competition with multiple sites; hence it is being viewed as one in this research context.

they are represented the most by far. Besides the large presence of Russian teams, the top 10 is mostly filled with countries coming from Latin America, such as Mexico, Chile and Brazil. This data is essentially a more detailed version of Table 1, as country and region are perfectly correlated (Cramer's V of 1) if you exclude the World Finals; all regions have unique countries, meaning that teams only participate in local competitions.

Country	Nr. of Universities	Country	Nr. of Universities
USA	552	PER	34
BRA	152	CUB	34
RUS	126	CAN	32
MEX	113	KAZ	29
BOL	68	FRA	29
COL	49	AUS	25
UKR	49	DEU	24
CHN	39	VEN	24
ARG	38	UZB	23
CHI	38	GBR	20

Table 3: Top 20 most represented countries by university⁵.

Looking at the representation of the universities as shown in Table 3, the order of the countries of the top 20 most frequent countries looks similar to the top 20 most represented countries by university. As this table again show absolute data, the strong influence of the number of participating teams per region must be kept in mind. The data distribution is again positively skewed where the USA is by far the most represented, as multiple North American competitions only have local American teams. Also, the universities from several Latin American countries are present in the top 10, meaning that either a relatively high amount of universities of this region are entering the competition or that these competitions have a relatively high number of participants. An interesting appearance is made by China, being represented by a significant number of 39 universities. Although they do not appear in the local competitions, they are well represented in the World Finals with a share of 13.45%; out of the 877 teams in all World Finals, 118 are from a Chinese university.

4.2.2.2 Problems and scores

In this section, all different parts of the data (as shown in the ERD of Figure 5) are described and summarized at a high level; at regional and/or competition level. In the subsequent chapter, a deeper look is taken, and the research questions are answered. This section is meant to give an overview of the data contents of each attribute and introduces all concepts analyzed later.

In Figure 8, the number of problems for each competition region per year are shown. This is a starting point for further analysis of the scores, as statistics should be relative to these number of problems to make them comparable. As the trend shows, the average number of problems

⁵ Note that these numbers represent the number of different universities *present in the data*, which is not necessarily the exact number of distinct universities; despite extensive efforts, due to differences in writing, the same university could appear multiple times written in a different way.

is slightly increasing, with the North American region having, on average, the lowest number of problems, and the Latin American and/or World Finals the most. The figure also shows that, besides some minor variations, the number of aggregated problems for each competition stays relatively consistent over the years. The number of problems does not necessarily say something about the difficulty of a year but does pose a somewhat different challenge. For example, it is known that the Latin American competition has relatively easier problems and (as shown for the latest years) have more of them. This is different from a European competition, where there are a fewer problems that are known to be relatively tougher.

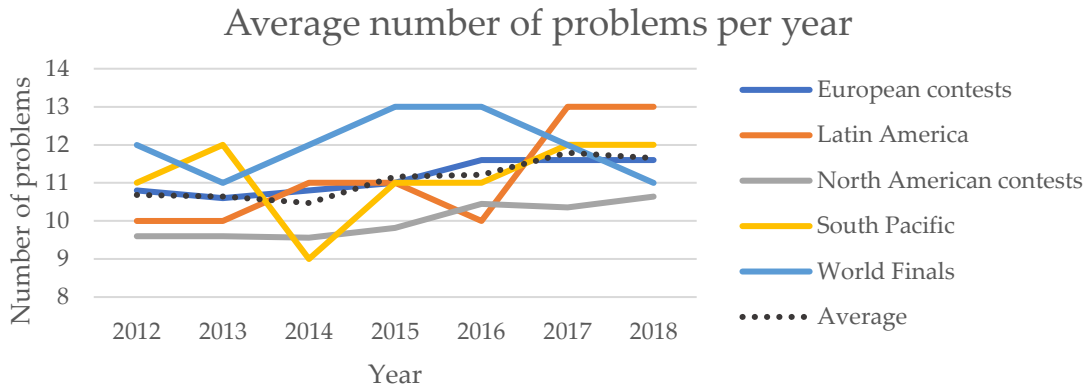


Figure 8: Average number of problems per competition per year.

With this influential statistic being clear, a deeper look can be taken into the scores. Figure 9 shows a boxplot with combined data for all years of the problems solved in each region, excluding teams that did not solve anything. It details the average, minimum and maximum percentage of problems solved, thereby indicating something about the difficulty of each competition for the teams that actively participated in it, and the general skill level of these teams. Keeping in mind that some competitions have more problems which makes it harder to solve all of them, this high-level overview shows that, on average, the Latin American competition has the lowest percentage of problems solved, while the South Pacific has the highest average percentage. As this data is very condensed, looking at the unique competitions in Figure 9b gives a more informative view of the (dis-)similarities between them.

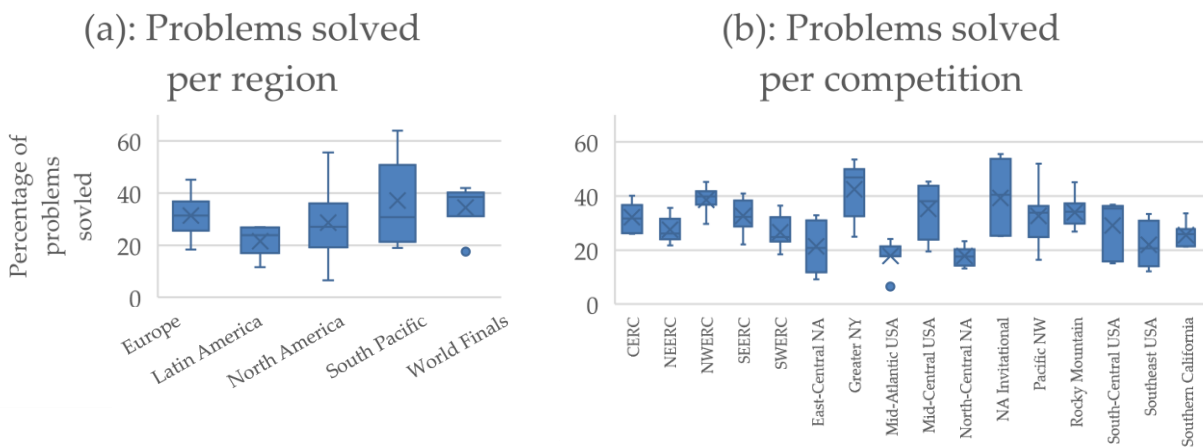


Figure 9: Problems solved, excluding teams without solves.

Noticeable differences can be seen here, were perhaps the most striking are the Greater New York-, North-Central NA- and Southern California- Regional Contest. Respectively, they have the largest average percentage of problems solved, lowest average solved and the tightest spread data. Furthermore, while these and the other North American contests are more different in this aggregated view, the European competitions seem to have more tightly spread data with competitions being relatively similar to each other. Also, note that the South Pacific has a small number of teams (sometimes around 12) and therefore has more fluctuations.

Looking deeper than only the number of solves and to get a complete view of the final total scores, the total time used for solving those problems must also be considered (note that this total includes the penalties for additional attempts). The data considered here is the average time spend (by active teams) working on each problem. As can be seen in Figure 10a and 10b, this number fluctuates more than the problems solved. When looking at the regions, the Latin American region appears to spend the least time per problem, and the South Pacific the most. The number and the size/difficulty of problems is likely an influential factor here. Teams competing in the Mid Atlantic USA Regional Contest take on average the least time per problem (27.21 minutes). In contrast, teams of the North American Invitational Regional Contest take the most time, roughly 56 minutes per problem. The difference in time spend is the least stable for the SWERC and East-Central USA Regional competitions.

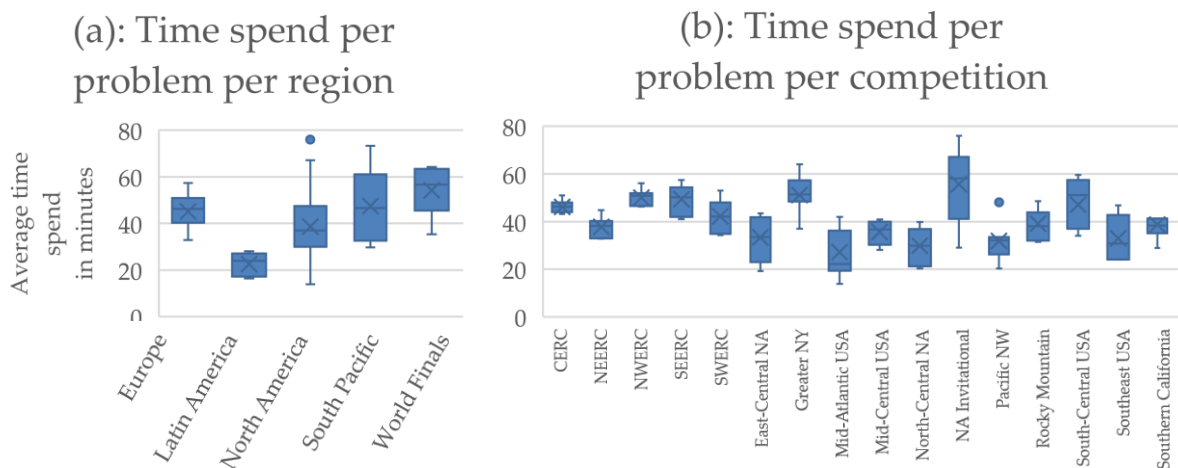


Figure 10: Average time spend per problem.

The time spend on- and the number of solutions per problem are of course related to the number of attempts problem. Figure 11 shows the average number of problems attempted per team for each competition. Numbers in this graph again represent efforts off all *active* teams, meaning that teams who attempted any amount of problems (but more than one) are included. This means that the average number is influenced by the overall performance of all teams, and as teams typically (on average) solve only a part of all problems, the average attempts are also lower. This high-level overview indicates that, on average, the number of attempts per problem fluctuates for all competitions, with a few extremes such as the Southeastern European Regional Contest (1.487 attempts per problem per team) and Latin America (0.6947 attempts per problem). Also, the World Finals appear to be more stable over the years.

Influential factors, the number of problems and the number of teams are the number of problems, are part of the calculation, but still play a role for the meaning of this average. For instance, Latin America has more problems in general, which is likely why a lower average number of problems is attempted. Furthermore, a large share of the teams typically solves only a small share of the problems, which lowers the overall average for all competitions, but exactly how large this share is, differs per competition. Finally, the (perceived) difficulty of the problems likely plays a role in the average attempts.

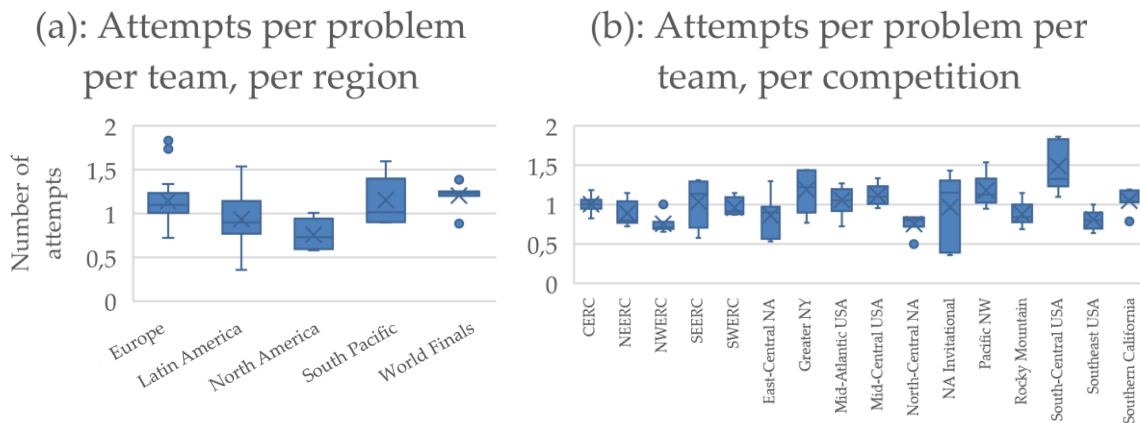


Figure 11: Average number of attempts per problem, by each team.

As the overall average attempts per problem are influenced by the average solutions per team, the way problems are solved matters. Figure 12 shows the (cumulative) distribution of problems solved. Here, it can be clearly seen that there is an inverse relation between number of problems solved and the number of teams which is expected, e.g. as the number of teams increases, there are less problems solved and vice versa. This relation should be obvious, but where the curve lowers indicates something about the average skill or average difficulty of each competition, where a higher proportion of solves on a higher number of problems means a higher average number of problems solved. Interesting enough, there are no major differences and all regions (on average) follow the same pattern.

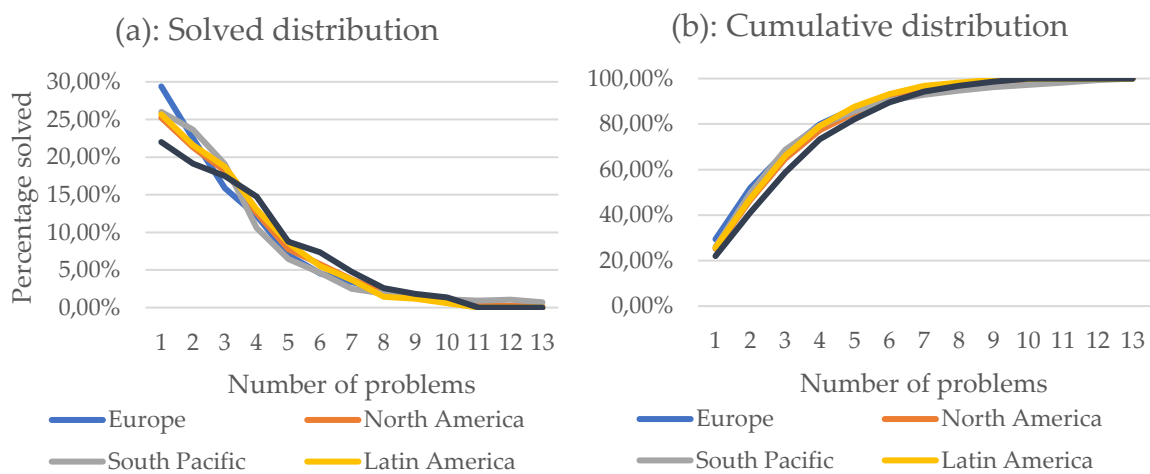


Figure 12: Distribution of problems solved over teams.

Roughly 40% of all solves are on two problems and 60% on just three problems. That they roughly follow the same pattern says something about the skill level of the teams, it could be that competitions have matched their problem set difficulty to the general skill level, making the curves similar despite overall differences between competitions. However, the World Finals seem to have a slightly different curve than the rest which is less steep; they have a smaller proportion of teams with lower percentages solved, meaning the general team in such a competition is capable of solving more problems.

This distribution has impact on the averages, but the average solution time per problem shown in Figure 13 is not affected and shows a whole different factor of the overall performance of teams. As it only looks at the time to solve, it implicitly excludes problems that were not solved and thus no time was spend on. Here, there can be clearly seen that the problems at the World Finals take, on average, one of the longest times to solve (over two hours), which is according to expectations as the World Finals is generally a difficult competition. In contrast, the teams of the Mid-Central USA Regional Competition only need roughly 90 minutes per problem. Latin American teams take a lover average time to solve and the NA region is on average above the European and South Pacific region in terms of higher solution times. Overall, there is more fluctuation and there are more differences between competitions. The same confounding variables, the number of problems and the skill level of teams, play a role here and the values might come from a combination of those two points. However, it is likely that the relation between these only exist because competition with more problems tend to have a greater number of relatively easier problems. For instance, as suggested by competition experts, the Latin-American problems are on average harder than many of the North American competitions. This makes sense, since the competitions are less important for the students in NA when compared to Latin America, where such competition may create opportunities for moving to another country and/or get a great job. Hence, teams in NA have less at stake in the competition and therefore might take it less serious in relative terms. The lower value from Latin-America might be influenced by that, but also by having an easier competition than in Europe (on average), though not easier than the World Finals (on average).

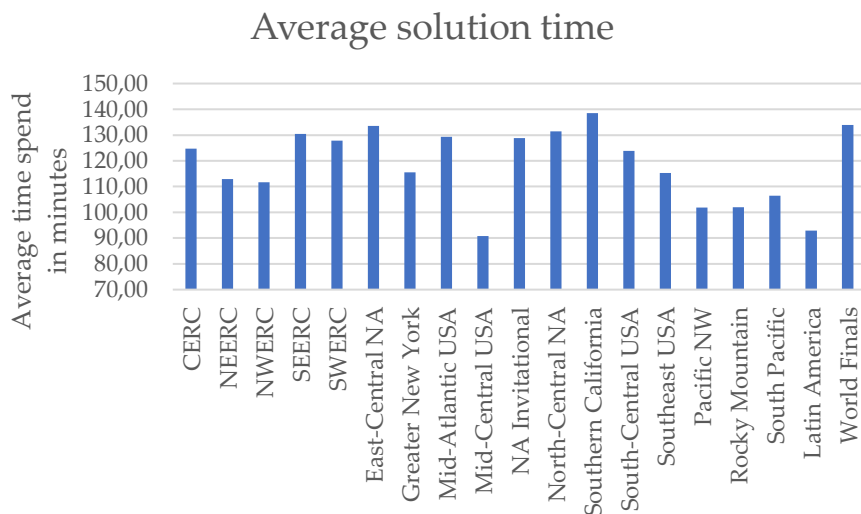


Figure 13: Average solution time for each competition.

Chapter 5: Descriptive and comparative analysis

This chapter focusses on describing and comparing all different parts of the data as introduced in the previous chapter. The goal here is to get a thorough understanding and overview of the past years, to become able to compare teams, problems and competitions. Where the data description investigated the data at a higher level, the approach taken here is to look deeper into the data, thereby answering the different research questions posed in Chapter 2 and finding explanations for statistics found. This iterative way of ‘funneling down’ starts by looking the general patterns in problems, followed by a descriptive summary and comparison of the regions and competitions, and ends with looking at top performing teams.

5.1 Concepts and general patterns

The purpose of this section is to introduce concepts used and detail general patterns found in the problem data. The approach taken is to first define measures and find a set of general patterns which can later be used for the description of each competition.

5.1.1 Comparative measures

A competition has a lot of characteristics and statistics that can be described and calculated. However, there are many aspects to consider, so a way for summarizing data must first be introduced, where the variables present limit the total information that can be derived and determine the way measures can be calculated.

When looking at the data, there are a couple of main themes, referred to as measures, that can be identified at a higher level. First, the *popularity* of a competition. This can be defined as a combination of the number of participating teams, the *absolute popularity*, and the background diversity of these participants (in terms of their country of origin and university), which tells something about how global the popularity is. This measure can be easily calculated in absolute numbers and summarize, but by adding some historic information a complete image and retrospective is provided. Then, there is the *difficulty* of a competition year, which is the direct result of the characteristics of a problem set posed. It can be measured in a lot of different ways, as it is a product of multiple factors; the number of problems (the *quantitative difficulty*), the hardness of solving each individual problem (the *qualitative difficulty*), and the distribution of difficulty over all problems in the set (the *overall difficulty*), e.g. are there many easy problems and only a small portion of hard problems, or vice versa. These measures can be calculated by the following metrics:

- The *quantitative difficulty* is based on the absolute number of problems in a competition.
- For the *qualitative difficulty*, the number of attempts and solves (avg, min, max) and their ratio, the proportion of teams that attempted/solved a problem, the time taken (avg, min, max) and the timepoint of solving (avg, min, max) play a role and are considered.
- The *overall difficulty* is not directly measurable, as no further information about the problem sets that are used is available at this point but can be partially derived from the distribution of solves over time.

- This shows not only the amount of solutions handed in and provides insights into the clustering of each problem’s solutions, but also shows the diversity over time of problems solved.
- Another factor here is the distribution of solutions/attempts over the teams, as it shows how hard the problems were for the average team.

Noteworthy is that indirect factors such as the skill level of the participants influence the results and the measure of *overall difficulty*. This skill level is a confounding variable that cannot be directly measured but must be considered as having an impact. The existence of these confounding variables and the lack of further data makes it hard to draw and generalize conclusions based on the derived information alone, which is the reason why the comparison and description will be done in a more qualitative and semantic manner.

5.1.2 General patterns in solution distributions

To get a more thorough understanding at why there are differences in the distribution of solves and thus the problem sets, an in-depth look must be taken at the overall difficulty of different years of all competitions. This can be done by looking at distribution of the number of solves per problem and adding the dimension of time. In these distributions, patterns can be seen which are characterized by a combination of timepoints, frequency and sparsity at which a problem is solved. Figure 14 introduces this way of plotting by showing the distribution of solutions for all problems in the dataset combined. A positively skewed distribution can be clearly seen here, where most problems are solved in the beginning of the competition.

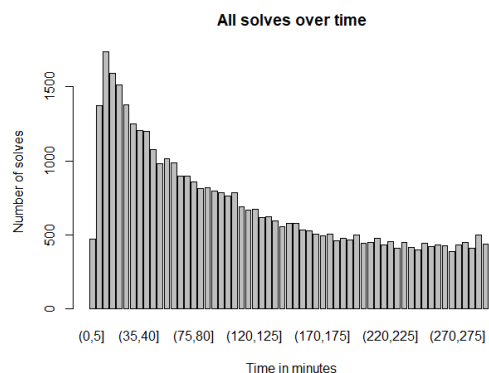


Figure 14: Distribution of all solutions over time.

More informative however is to look at the individual patterns found in the data for each problem. Solution distribution graphs are looked at and found patterns are described and illustrated in Appendix B. Looking at these patterns, there are seven general patterns found in the solution distributions of the problems⁶. In general, the found solution patterns seem to concentrate at either the beginning or end of the competition or have no concentration at all. The most occurring pattern is that of problems that have solutions handed in over all time points, with no specific clustering or peak. This is what you would expect, as the choice of the order of working on each problem differs, and the skill level of each individual team

⁶ These are the most outstanding ones, but it is not necessarily a full exclusive list; there could be more patterns not seen at this point in time.

determines how fast a solution is found making the solutions more spread out. In the data, this translates into a situation where, when a solution is found, solutions are being handed in sporadically until the end of the competition which is seen in every competition and year. The found patterns and thus structure in the data can be used to characterize problems individually, and in turn say something about the problem set as a whole for competitions and years.

Going deeper in the general problem patterns, there is looked at the order of the problems in the set. The order of difficulty in which the problems are listed is usually random to have the teams figure out by themselves which problems are doable. To verify if the order is indeed not important, Figure 15 shows the distribution of solves and the time spend on each problem. The problem that is solved most is problem A with a support of 0.5865, which indicates that teams most often attempt and solve the first problem they encounter. The figure supports this statement, as it shows that the average timepoint of solving is lower for problem A. On the average actual time spend for each problem however, problem A is not the lowest (51.31 minutes, whereas problem L has 46.16 and M 48.14), although the average standard deviation is 54.011 minutes. Another interesting observation is that problem M (keep in mind that this is the 13th problem, and a lot of competitions do not have that many problems) has the least amount of solves, but also the least average amount of attempts (1.814). A reason for this could be that the authors of the problem sets sometimes tend to put the harder problems at the end, which is known amongst the teams, or because it is simply the last problem teams read when going through the set, so they leave this problem for later or not attempt it at all.

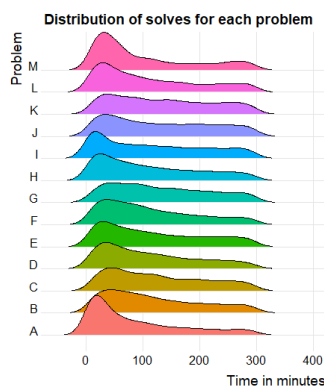


Figure 15: Overall distribution of problem solutions.

5.2 Competition descriptions

In this section, each of the regions and their respective competitions in the dataset are described and characterized. Each summary consists out of filling in the concepts defined in previous chapter and a description of the visualizations per year.

5.2.1 World Finals

The World Finals is the biggest competition of all competitions present in the dataset and in the world. It is the competition where all teams end up in, if they perform well enough in their local competitions and advance through. Being the most high-profile and most prestigious of all, it is also viewed as the competition which has the highest interest of the general public.

Popularity

Over the past years, the size of the World Finals has grown. In contrast to other competitions however, there is a fixed number of slots available each year. This number is defined by the organization, and so the number of teams in this sense is not defined by the popularity directly. The increase in slots does represent the view of the organizers that the competition became more popular, as they want to accommodate more teams each year. As shown in Table 4, the number of teams and diversity of their background has been steadily increasing.

Year	Nr. of teams	Nr. of universities	Nr. of countries
2012	112	110	37
2013	119	118	36
2014	122	121	45
2015	128	127	39
2016	128	127	40
2017	128	127	43
2018	140	137	50

Table 4: Popularity of the World Finals.

Over the past seven years, teams from 66 nations and 336 distinct universities entered the World Finals.

Difficulty

Quantitative difficulty

In terms of quantitative difficulty, there are an average of 12.00 problems posed to the contestants each year. Of these 12 problems, there are usually one or two problems not solved at all, and the distribution of the overall difficulty for the remaining problems varies. As shown in Figure 16 below, there is usually one or a couple of problems that are solved by the majority, a set of problems solved by around a quarter of the contestant and a couple of problems only solved by a small portion of the teams. The overall and average height of the bars per year says something about the difficulty of the problems, where can be seen that 2014 was a particularly difficult year, which is likely because of its problem set. It looks like this was compensated for in the next year(s), because of the peaks and overall higher height of the bars per problem.

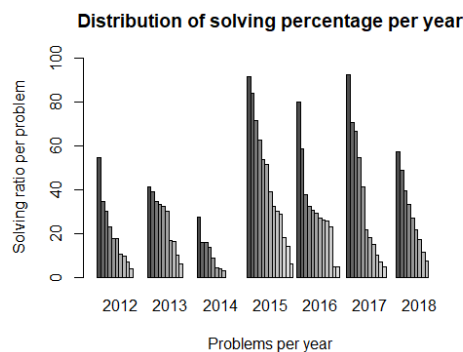


Figure 16: Distribution of solving ratio for the World Finals.

Overall difficulty

The World Finals are known for being one of the hardest competitions. This is no surprise, as the aim is to find the best of the best. Looking at the overall difficulty is therefore always in relative sense, as the expected skill level of teams in this competition is higher.

Shown in Appendix C are the solution distributions over time for each year separately. The overall amount of solves from previous Figure 16 combined with this information shows that there is clearly a process and multi-year trend of problem balancing going on. Up till 2014, there was a decreasing number of solutions, after which there was a big jump and decrease again. It took the organization a couple of years to take the average difficulty to the desired level. This desired level is reflected in their goal to ‘make all teams solve one problem, but no team solve everything’. The distribution of 2018 seems to be the desired one, as it seems to have problems of all difficulty levels.

Looking at the individual problem patterns that occur over the years, there seem to be one or more easier problems identified and solved in the beginning. Apart from 2014, all years show a peak or cluster of solutions at the start, after which multiple problems are solved sporadically throughout the whole duration of each match. Remarkable is that there is a consistent trend of almost all problem solutions being spread and not clustered around a certain time. There are some low number of solutions that only occur at the end, but problems with a higher amount of solutions all appear throughout the competition. The likely cause of this pattern is probably the difference in skill level between teams.

5.2.2 Europe

The European region consists out of five competitions, divided by geographical region.

Popularity

There are differences in the popularity of the European competitions. First, where the Central, North and Southeast competitions show a stable number of teams, the Southwest and Northwest show a growth, which might indicate an increase in interest in the competition over the more recent years. There has not been a change in the eligibility rules, so increased popularity is the most likely reason. This growth is combined with a larger diversity of universities. The number of countries however remains stable across all competitions.

	CERC			NEERC			NWERC			SEERC			SWERC		
	Team	Uni	Ctry	Team	Uni	Ctry	Team	Uni	Ctry	Team	Uni	Ctry	Team	Uni	Ctry
2012	77	31	6	77	31	6	82	43	10	77	43	9	43	26	6
2013	72	30	6	72	30	6	92	46	10	79	48	8	43	23	6
2014	78	33	7	78	33	7	95	48	9	74	39	7	49	27	6
2015	61	26	7	61	26	7	95	51	10	84	40	6	52	30	6
2016	66	27	6	66	27	6	114	60	10	81	43	6	60	30	6
2017	69	31	7	69	31	7	118	60	10	79	44	7	75	48	6
2018	73	32	7	73	32	7	118	57	10	85	44	9	89	47	6

Table 5: Popularity of European competitions.

Combined for all five competitions, teams from 46 nations and 543 distinct universities entered in European competitions. Each of those nations performs differently in their respective competitions, as shown in Figure 17. Relatively for each competition (in order of the figures below from left to right), English & Latvian, Finish & Icelandic, Polish & Croatian, Ukrainian & part of Bulgarian and Swiss & part of Israeli teams are the relative highest performers, with respect to the number of teams (e.g. the Russian teams in NEERC often end up in the top ten, but have a lot more teams competing).

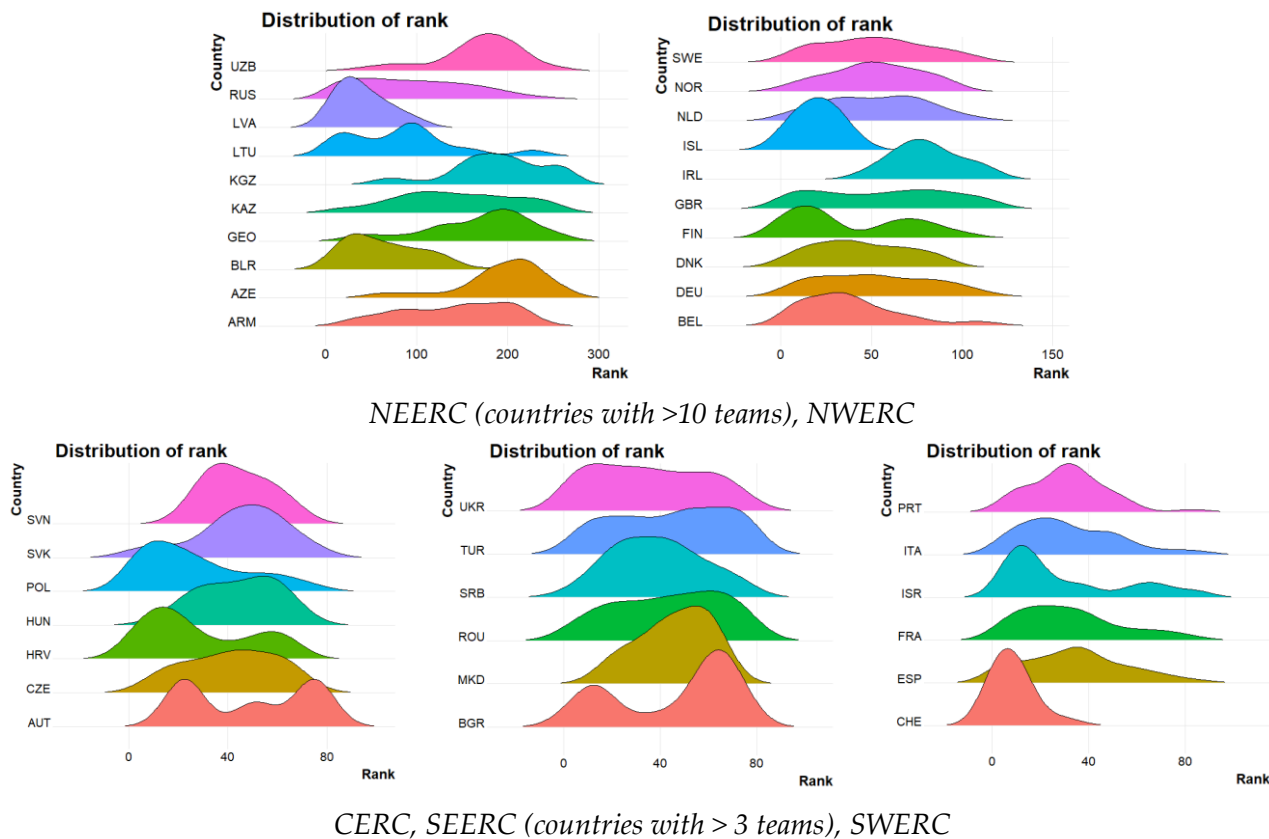
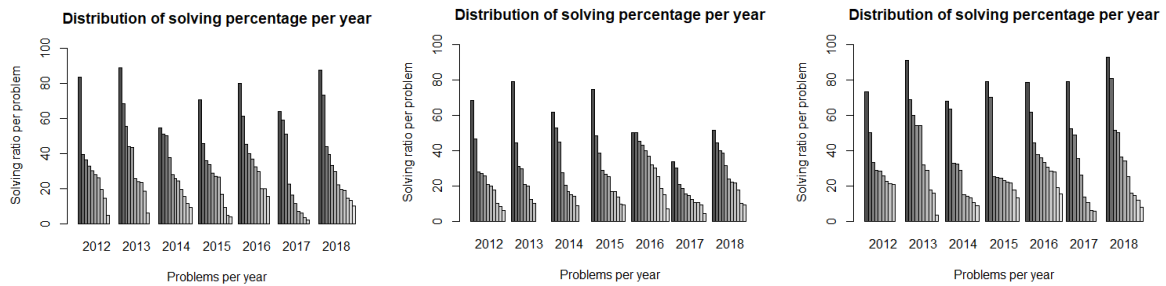


Figure 17: Rank distributions for each country per competition.

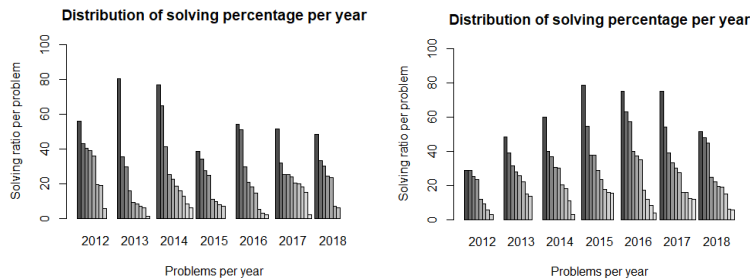
Difficulty

Quantitative difficulty

In terms of quantitative difficulty, there are an average of 11.143 (std of 0.879) problems posed to the contestants of the European region each year. For all these competitions and years, there is almost always at least one problem not solved, with the SWERC almost consistently having one not solved problem (std=0.378) and the SEERC having the most fluctuation (std=0.9589). Looking at the distribution of the problems solved over the years, the CERC and NWERC are the most stable in terms of problem-solving ratios across years. Most competitions and years follow a similar pattern and show one or a couple of problems that are always solved much, a larger set of problems solved by around 20 till 40 percent and some problems that are solved by less than 20 percent. An interesting year is 2016 for the NEERC, where the problem difficulty seems to be really well spread out, with each problem only being solved slightly less than the other.



CERC, NEERC and NWERC



SEERC and SWERC

Figure 18: Distribution of solving ratio per year for Europe.

Overall difficulty

As the European region consists out of multiple competitions, individual graphs for each year are placed in Appendix C. Some of the European competitions are more similar than others, but simple aggregation of the overall difficulty graphs is not possible. Therefore, each competition is summarized separately in order below.

CERC: The Central European competition used to have one or two problems being solved fast by a relatively high number of teams, but from 2017 this pattern no longer occurs. There even seems to have been a gradual change from 2015, with a high peak of solutions no longer occurring in the later years, and the problem solutions being more spread out. The diversity of the problems attempted over time seems to be consistent. A noteworthy year is 2018, as there are gaps visible where no solutions are handed in. It appears that there were technical difficulties with the computers that year⁷, so teams had no opportunity to submit solutions in those timeframes.

NEERC: The Northeast European contest seems to be quite consistent in terms of similarity of the solution graphs for each year, so it can be characterized by several patterns. First, there are always (except in 2016) one or two problems solved fast and by many teams. After this peak, there are usually around two problems with a peak of solves around one-third of the time, and there are some problems being solved throughout the competition. Remarkable is the clustering of solutions, where solutions occur (almost) exclusively around a certain timepoint or solutions are continuously being handed in. This might indicate that teams of this competition use the real-time scoreboards to a higher extent.

⁷ <http://codeforces.com/blog/entry/63606>

NWERC: The Northwest European contest, like many others, also shows an almost consistent peak of solutions at the start of each year. Especially in the later years, these peaks are higher. Furthermore, with a rise in teams, the patterns less visible in the earlier years seem to be amplified in later years. Another pattern is that some of the more sporadically solved problems are spread over the entire match. In addition, apart from the beginning, solutions are not often clustered.

SEERC: The first thing noticeable about the Southeast European contest is the relatively high diversity of solutions as many different problems are solved, and the overall amount of solves. This overall amount is relatively low, as also could be seen in the distribution of solving ratios in previous Figure 18. Especially 2015 and 2018 show an overall lower number of solves. There appears to be a trend toward the solutions being more clustered over the years.

SWERC: The rising arc in the distribution of problems solved for the Southwest European contest is also visible in the difficulty graphs. Where the earlier years show only few solutions, more recent years show much more filled figures. The competition doubling in size from 2012 till 2018 plays a big role here. This change makes the older years less representative for the current situation, but older patterns are that problems are solved throughout the competition with no clear peaks, solutions are spread out and (likely due to a lower number of teams) diversity of problems solved is higher. The more recent years show a trend where there is at least one problem being solved often in the beginning, and continuation of none of the problem's solutions being clustered around a certain time.

5.2.3 Latin America

After advancing from the national qualifiers, teams in Latin America end up at the regional finals. The competitions here are a bit different than the others. Although at different sites, they all occur simultaneously and use the same problem set but are still seen as distinct competitions by the ICPC. However, as all other competitions use unique problems for each year, all Latin American competitions are aggregated into one for a fairer comparison.

Popularity

The Latin American competitions have been undergoing some changes in the beginning, as can be seen in the big increase of teams in 2013, shown in Table 6. Since then, the popularity of the competition seemed to have remained fairly stable across the years, with an overall trend of the countries, universities and teams staying the same.

Year	Nr. of teams	Nr. of universities	Nr. of countries
2012	171	114	5
2013	497	259	10
2014	402	237	13
2015	412	237	13
2016	400	238	13
2017	434	262	13
2018	398	252	12

Table 6: Popularity of Latin America.

The past few years, teams from 15 nations and 565 distinct universities entered the Latin American competition. Of these nations, the Brazilian teams were the most successful, but Cuban teams also do well, with the highest average placement and a large portion of the entered teams ending up relatively high. Figure 19 visually shows this in the form of distribution of ranks for each country.

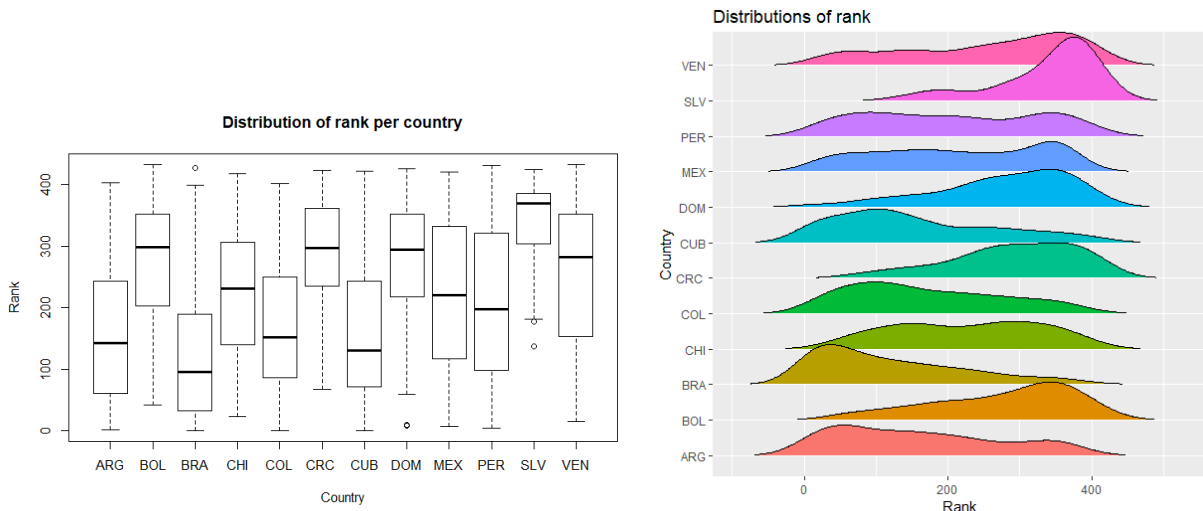


Figure 19: Distribution of rank for each country (with appearance in more than one year).

Difficulty

Quantitative difficulty

There are an average number of 11.143 problems posed to the contestants each year, with the more recent years having 13 problems. It fluctuates per year how many problems remain unsolved, but on average there is around one problem not solved. The distribution of the solving percentage shown in Figure 20 indicates that there always are one or a small group of relatively easier problems, which are solved more often and have solving ratios above 50%. There is a fairly stable average of solving percentages over the years going on, but 2013 seemed to be harder than other years. Furthermore, with a larger problem set as of 2017, the solving ratios are more spread out instead of being more equally divided between problems.

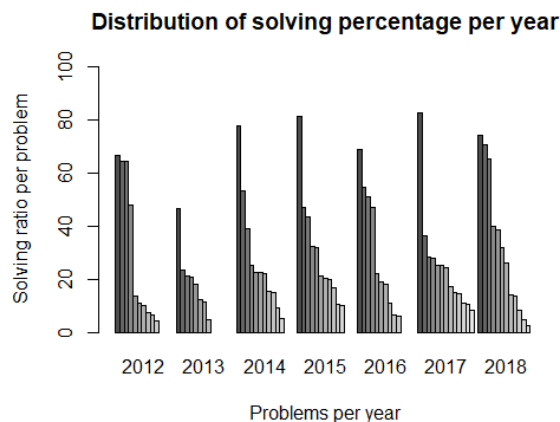


Figure 20: Distribution of solving ratio per year for Latin America.

Overall difficulty

As the overall number of teams aggregated into this dataset is high, the distribution plots shown in Appendix C are more informative. This also has an effect on the total solutions handed in and thus the scale of the figures, which must be kept in mind when interpreting results.

The distribution graphs clearly all follow two very similar patterns, where there is a large number of solutions for the same one or two problems in the beginning, and one problem solved many times throughout the whole duration of the competition. As also can be seen in the previous distribution figure, the year 2013 is the most different from all others, in the sense that it follows more or less the same pattern, but the ‘easier’ yellow problem in the beginning took the teams more time. The years 2012 and 2016, and 2014 and 2017 show the most similarity for the timeframes in which the easier problems are solved. Furthermore, 2018 is a very interesting year as there are multiple problems identified fast which can be solved by many teams, but it takes them more time. A final noteworthy note is that in 2016, a problem was disqualified and removed.

5.2.4 South Pacific

The South Pacific region is distinctly different from the others, in the sense that there are a lot fewer teams participating each year. Its geographical location also makes it different, as there are only two countries present in that region.

Popularity

The popularity table of the South Pacific (Table 7) shows a big dent in the earlier years, and a steady number of teams after that. The reason for this fluctuation is because of changes in the official rules, which happened multiple times the past years. Starting in 2014, the regional finales became invite only, meaning that only teams in the top 4 of their division and the best of their university are eligible, or when a wildcard position is handed out. However, recent alterations to the rules have led to more changes; the unique university constraint was lifted (now two teams from the same university can be invited) and only maximally two universities of each division are invited⁸. Looking at the countries, only two countries (Australia and New Zealand) compete and are both represented each year.

Year	Nr. of teams	Nr. of universities	Nr. of countries
2012	86	25	2
2013	88	25	2
2014	13	12	2
2015	12	12	2
2016	12	9	2
2017	10	7	2
2018	12	10	2

Table 7: Popularity of the South Pacific.

⁸ <https://sppregional.org/rules/>

In total, teams from 32 distinct universities entered the competition, where the unique number of universities was dependent on the performance of the teams.

Difficulty

Quantitative difficulty

On average, there are 11.36 problems posed to the contestants each year. Except for 2014, the total number of problems has always been 11 or 12. Of these problems, there were two or three not solved at all up till 2015, after the difficulty of the problem set likely altered because only one or none were not solved from 2016. This can also be seen in the distribution of the problems solved, where there is a big increase in (average) solving percentage starting in 2016. There are also more problems being solved more often. However, there must be kept in mind that there are only few teams in the more recent years, which have a relatively big impact on the solving ratios. For example, if just one additional team in 2017 solved the easiest problem, the solving percentage would go up by 10%. Overall, a trend towards more problems being solved per team can be seen.

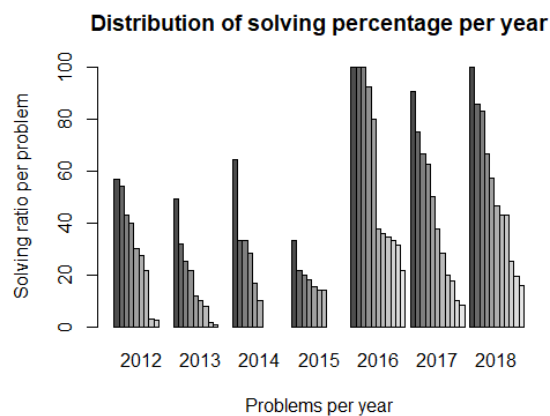


Figure 21: Distribution of solving ratio for the South Pacific.

Overall difficulty

As a first note, the distribution graphs given in Appendix C (or any visualization of distribution for that matter) become less informative when the number of teams and thus solutions are lower. This also creates a different challenge for the teams; there is much less information on the scoreboards to look at and learn about the difficulty of the problems. This must be kept in mind when interpreting data and results.

In the first years with the larger number of teams, you can clearly see that there are one (in 2012) or two (2013) problems solved relatively fast by a large share of the teams. The other solved problems are more spread out. However, starting from 2014, the situation changes. The overall amount of solutions is very much lower, but the diversity of the problems solved grows each year, which is what you would expect when teams attempt problems at their own insight; they have no scoreboard to look at so they need to identify the most solvable problems themselves. With the exception of 2014 and 2015, the more recent years also show an easy problem being solved fast. The rest of the distributions do not really contain specific patterns.

5.2.5 North America

The North American region is the largest in terms of number of teams and competitions. It consists out of thirteen competitions, from which eleven are present in the dataset.

Popularity

The absolute popularity for each competition differs, most likely also due to geographical location. In below Table 8 the countries are omitted as teams originate from either the US, or the US and Canada.

	East-Central NA		Greater NY		Mid-Atlantic USA		NA Invitational		North-Central USA		Pacific NW	
	Team	Uni	Team	Uni	Team	Uni	Team	Uni	Team	Uni	Team	Uni
2012	122	63	41	21	165	62	24	24	194	60	111	34
2013	121	61	51	21	183	62	23	23	228	64	112	32
2014	122	59	45	17	160	56	21	21	245	64	88	32
2015	122	55	49	18	178	60	45	45	180	60	69	33
2016	123	48	48	19	153	55	44	44	215	63	73	33
2017	138	47	53	22	162	57	46	46	185	58	68	32
2018	133	49	68	25	168	56	43	43	198	59	61	26

	Rocky Mountain		South-Central USA		Southeast USA		Southern California		Mid-Central USA	
	Team	Uni	Team	Uni	Team	Uni	Team	Uni	Team	Uni
2012	47	16	53	27	56	15	74	25	<i>n/a</i>	<i>n/a</i>
2013	37	14	58	27	46	15	91	31	<i>n/a</i>	<i>n/a</i>
2014	53	17	60	23	<i>n/a</i>	<i>n/a</i>	82	25	<i>n/a</i>	<i>n/a</i>
2015	52	17	60	24	42	14	87	26	149	64
2016	54	16	67	26	45	12	85	27	150	56
2017	52	13	74	28	46	13	95	31	119	49
2018	63	16	71	29	33	9	86	26	120	49

Table 8: Popularity of North American competitions.

In terms of absolute size, the North-Central USA is the biggest competition, while the Southeast USA is the smallest. The NA invitational is the most diverse, with all teams coming from a different university. This is due to the fact that only the teams that were in the top five schools from each respective ICPC region at that time were invited⁹.

Clearly, the competitions differ in multiple ways. There are different patterns going on in the popularity of the competitions. First of all, there are a few competitions for which the number of competing teams remains fairly stable, these are the Southern California, North-Central USA, Rocky Mountain, Southeast USA and Mid-Atlantic USA. Then there are some who show a steady growth; the East-Central NA (with the number of universities declining), the Greater NY and NA Invitational. Finally, the popularity of the Pacific NW seems to be declining. For the last competition, the Mid-Central USA, there simply is not enough data to see a trend. Combined, teams coming from 577 different universities and originating from the US and/or Canada have entered in the North American competitions.

⁹ <http://naipc.uchicago.edu/2018/>

Difficulty

Quantitative difficulty

In terms of quantitative difficulty, there are an average of 10.03 (std of 1.29) problems posed to the contestants of the North American region each year. While most competition fluctuate around a problem set size of 10, the Pacific NW has an average of 12.29 problems each year and the Mid-Atlantic USA has a constant size of 8.

It is dependent on the competition how many problems are unsolved each year. Competitions such as the East-Central NA and Greater NY almost always have all problems solved at least once, while the Mid-Atlantic USA (std of 0.95) and Southeast USA (std of 1.05) have around two problems unsolved. The overall average lies around one unsolved problem.

Looking at the distribution of the problems solved over the years, first of all the Greater NY stands out as it has high solving ratios for almost all problems solved in all years. In contrast, the Mid-Atlantic USA has much lower ratios and also less problems solved, but this is also due to the fact that there are only eight problems each year. East-Central NA and North-Central USA appear similar, with both having more solved problems with an overall lower solve ratio. Also, the Southern California and Rocky Mountain show similarities in terms of (a few) peaks and overall distribution per year; both are quite stable across years. The Pacific NW interestingly enough seems to show a trend toward a higher average solving percentage, which could be due to the fact that the fewer contestants are more skilled, or the organisation tries to make the competition more appealing by posing easier problems. Finally, for the Southeast USA and Mid-Central USA there is some missing data, which makes it harder to see trends. However, 2012 and 2016 respectively seem to have been more difficult years.

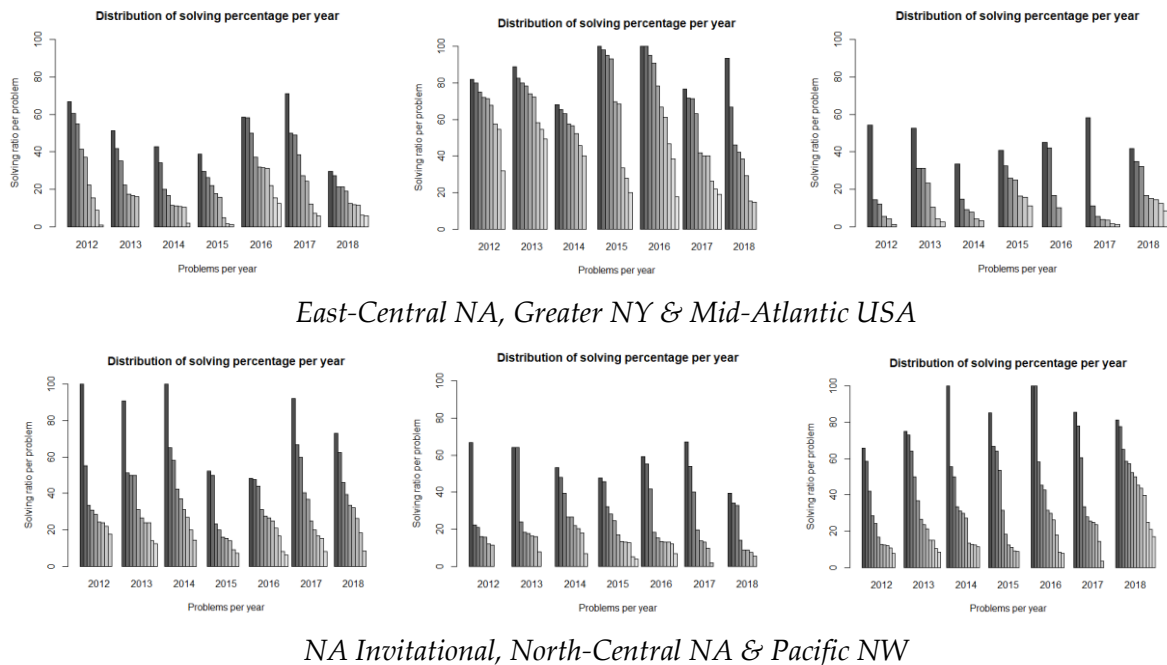
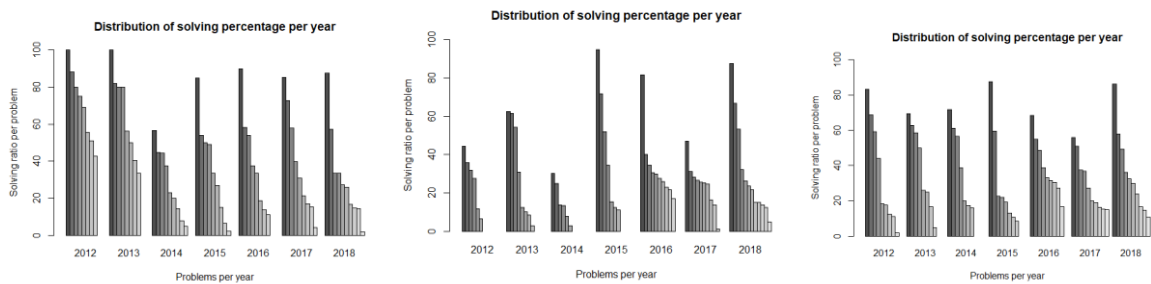
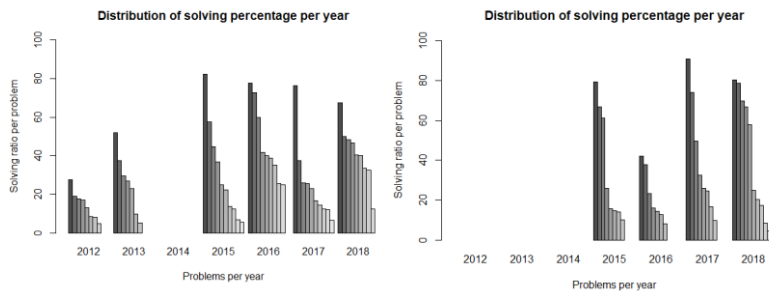


Figure 22(1): Distribution of solving ratio for North America.



Rocky Mountain, South-Central & Southern California



Southeast USA, Mid-Central USA

Figure 22(2): Distribution of solving ratio for North America.

Overall difficulty

With eleven competitions being present, figures of each year for those competitions are placed in Appendix C. General descriptions are given and the most noteworthy and remarkable trends are summarized below.

East-Central NA: Although the East-Central NA regional contest has over a hundred competing teams each year, the number of solutions and thus the average solving percentage is relatively low for most years. There appears to have been significant differences in the difficulty of the problemsets, as it is less likely that the skill level of all teams would fluctuate that much. An argument for this statement is the fact that the graphs show no consistent patterns over the years, where only some years start with easy problems, the diversity of solutions is different for each year and the overall number of solutions handed in differs a lot, with a possible trend from less to more solutions (with the exception of 2018).

Greater NY: As could also be seen in Figure 22, the overall solving percentage in the Greater NY regional contest is high. This is also reflected in the diversity of problems solved. The overall number of solutions is therefore relatively high for the total number of teams. Two patterns stand out; first there always is a problem, which is always problem A, solved more in the beginning and second, the rest of the solutions is mostly spread out over the entire duration. A noteworthy year is 2017, where a big gap can be spotted. Although no news can be found about this, this probably is due to technical difficulty. The latest solutions are also handed in after the 300 minute mark, which indicates additional overtime that normally is only added to compensate for earlier time loss.

Mid-Atlantic USA: Although the Mid-Atlantic USA regional contest is one of the bigger competitions in North America, the average solving percentage lies fairly low. Especially 2014

and 2016 seemed to have been hard years, as the overall number of solves and diversity of problems solutions are low. This low diversity can be the reason that (also in other years), there are relatively fewer solutions. Remarkable is that in all years, problem A was one of the solvable problems and most years, it was an easier problem being solved much. No overall yearly trend can be seen, as there is a lot of variation.

NA Invitational: As the name suggests, the NA Invitational regional contest only has a selected number of teams competing each year, making it one of the smaller competitions. The solving ratios in combination with a relatively high diversity of solves that are unclustered however show that the skill level of the teams is likely higher than average. There seems to be a trend of adding an easier problem which is solved faster from 2015 onwards, with high clustered peaks in 2017 and 2018. Another note is that in 2017, the overall number of solves is relatively low.

North-Central NA: The North-Central NA regional contest is the biggest in this dataset, with around twohundred teams entering each year. With more teams, the expected average skill level is lower, which can be confirmed by looking at the solving ratios. Besides there often being a problem solved a lot in the first half of each year (except for 2015), in the earlier years, there is also a second problem solved in parallel. The more recent years show a peak at the beginning, a more often solved but scattered problem (or problems) after this and some less structured solutions.

Pacific NW: The solution distributions for the Pacific NW regional contest mostly seem to follow the same pattern; a peak of multiple solutions being handed in at the beginning (one or two problems solved a lot with a secondary wave of solutions for another problem) and more scattered solutions after that. Although 2016 seems to have been a different type of year (due to a harder problemset or less skilled teams), the declining interest in the competition cannot really be seen in these graphs. Interestingly enough, the organization themselves state that the first division (which is considered) has a hard problemset¹⁰, which means that teams must overall be more skillfull as the both the number of solutions and average solving percentages are relatively higher. Finally, a gap at the end of 2017 with unknown reason can be spotted.

Rocky Mountain: With an above average solving percentage, teams in the Rocky Mountain regional contest manage to solve the relatively fewer number of solutions more efficiently. There is a trend towards more solutions being handed in total, meaning that the difficulty is decreasing or the overall skill level of the teams is growing. Remarkably enough, problem A is always solved in high amounts at the beginning (except in 2013, where it's solves are scattered). The other problems attempted are more spread out however.

South-Central USA: Although the number of teams entering in the South-Central regional contest is fairly stable, there are big differences in the number of solutions handed in each year. The earlier years 2012 and 2014 were probably a lot more difficult, while the more recent years show a greater number of solutions handed in. Recent years show a trend towards

¹⁰ <http://www.acmicpc-pacnw.org/>

easier problems being solved fast in higher amounts, but most solutions are unclustered and spread over time. At the end, some new problems often appear on the timeline, increasing the overall diversity and indicating why the average solving percentage is relatively low.

Southern California: With a balanced average solving percentage each year, the Southern California regional contest shows similar solution distributions over time. There are some smaller clusters in the beginning, but no consistent timepoints with significantly higher peaks. Also, there seems to be a multi-year trend along the entire timeframe of two problems being solved in high amounts for the entire duration of the match. The competition seems to have equally balanced the difficulty of the problem sets in the past years. A noteworthy year is 2017, which has taken an hour longer (due to unknown reasons).

Southeast USA: The Southeast USA regional contest is among the smaller competitions, which is reflected in the solution distributions. Although the solving percentage is around average, there are not many solves for each problem. The diversity of problems however is higher, and the distribution of their solves is scattered. There is always one problem solved (relatively) more in the beginning, but there are no real clusters besides this. Likely due to its small size, no general trends can be spotted.

Mid-Central USA: For the Mid-Central USA regional contest, the first few years are missing, but the more recent years show similar distributions. The overall solving percentages are high, which is due to the fact that there are (except in 2016) multiple easier problems solved at the start, followed by a problem solved in medium amounts. Due to missing years however, it is difficult to see trends.

5.3 Competition comparison

With all competitions summarized in previous section, they can now be compared on the metrics (defined in section 5.1.1) and averages of the numbers and figures shown before.

Previous section has showed that there are some differences within competitions. With some competitions being internally more consistent (e.g. NEERC), others show a lot of variance (e.g. Mid-Atlantic USA). This section looks beyond the internal trends and compares on the variance between competitions. However, a baseline to which everything can be compared against must be established first. This baseline will be the World finals, as the serious nature of this competition is a good representation of the skill level that local competitions must prepare their teams for. The primary goal of the World Finals is to find best the best of the best programmers; thus, their focus is also on making their problem set difficult to solve. They aim to add one easier problem (as mentioned before, this is stated in the mission of the ICPC), but the rest of the problems are known for being hard. In contrast, local competitions are known for having a reason to monitor the difficulty of their problem sets; they need to match it with the general skill level of the competing teams to keep their competition fun and attractive. With this information in mind, and the World Finals being the most important and prestigious competition, those statistics will be used as the starting point for comparison and as baseline.

5.3.1 Problem set comparison

Competitions can be characterized by a set of numbers, representing key and defining attributes for each of them. Below Table 9 shows the average solving percentage over all problems and the percentage of problems solved by each team. These two numbers give an indication of the average difficulty of the problem sets and skill level of the average team, respectively.

Competition	Average solving% per problem	Average % of problems solved per team	Competition	Average solving% per problem	Average % of problems solved per team
World Finals	30,00%	34,84%	East-Central NA	25,78%	21,64%
Latin America	28,24%	22,55%	Greater New York	60,27%	40,94%
South Pacific	38,76%	28,94%	Mid-Atlantic USA	19,10%	17,96%
CERC	32,04%	32,15%	Mid-Central USA	35,28%	34,21%
NEERC	27,08%	27,56%	NA Invitational	34,86%	33,52%
NWERC	35,00%	38,43%	North-Central NA	24,83%	19,30%
SEERC	24,74%	31,03%	Pacific NW	37,56%	27,12%
SWERC	28,70%	27,60%	Rocky Mountain	43,14%	32,27%
			South-Central USA	29,30%	32,88%
			Southeast USA	31,02%	24,64%
			Southern California	35,36%	26,40%

Table 9: Measures for each competition.

The Greater New York regional contest stands out with having the highest percentage for both metrics and the Mid-Atlantic USA regional contest with having the lowest percentages. Most European competitions and Latin America show values for the average solving percentages and problems solved per team below the World Finals, but the North American competitions fluctuate. This probably means that the average skill level of these teams is mostly below the World Final standard. The difficulty of most problems in the NA region is also below the baseline, as the problems are solved with higher solving ratios.

Looking further at the problems, the average solution time for each problem explains part of its difficulty. If the time is higher, this means a problem is more often solved later during a competition or that it is solved exclusively at the end. Problem sets are built to give teams of all skill levels something to do all the time, so a spread of these means over the full 300-minute time period is what is aimed to be achieved. Combining the means of all problems for each competition results in the average time at which all problems are solved, which is shown in Table 10. The World Finals have the highest mean and second highest median (194.71 minutes, only the NEERC has a higher median of 199.13 minutes) with the smallest standard deviation of 56.01 out of all competitions, and also the biggest timepoint including the average penalty (calculated by the average attempts times 20 minutes). The deviation has likely to do with the difference in skill-level of the teams, as a larger value means that some teams were able to solve a problem quick, and other took more time. This means that the problems in the World

Finals are most often solved later but with a smaller spread, which indicates that most teams take longer to solve problems and thus that the competition is relatively harder. This is also shown by the fact that there is a large portion of time added for the additional attempts taken. European competitions also show higher means, but more fluctuations in the standard deviation. There is more time included for penalties however, with the SEERC (69.68 minutes) and NEERC (60.61 minutes) having the highest average penalty included.

Competition	Mean timepoint of solving	Mean timepoint incl. avg penalty
Mid-Central USA	135,35	179,09
Greater New York	148,36	178,83
South Pacific	152,53	198,25
Pacific NW	153,36	194,24
Rocky Mountain	153,83	199,26
Southeast USA	161,08	206,68
South-Central USA	165,12	215,59
NWERC	166,66	212,70
Southern California	168,66	207,69

Competition	Mean timepoint of solving	Mean timepoint incl. avg penalty
NA Invitational	168,88	219,96
Mid-Atlantic USA	170,41	223,80
SEERC	172,63	242,31
North-Central NA	172,64	222,43
Latin America	173,37	216,92
NEERC	174,61	235,22
CERC	175,08	224,54
SWERC	176,46	226,89
East-Central NA	179,05	228,62
World Finals	185,16	244,34

Table 10: Timepoint of solving per competition.

5.3.2 Country comparison

To see which competitions (and thus countries) prepare their teams the best for the World Finals, Figure 23 shows the distribution of the rank for each country. Although some countries are represented way more than others (e.g. Russia 142 times and Mexico 20 times), the best performing countries are the Polish, Belarusian and Russian teams as they most often on average end up in a high spot. So apparently, Eastern European countries do well. This conclusion is in line with the local competition distributions, where these countries also do relatively well in their own competitions.

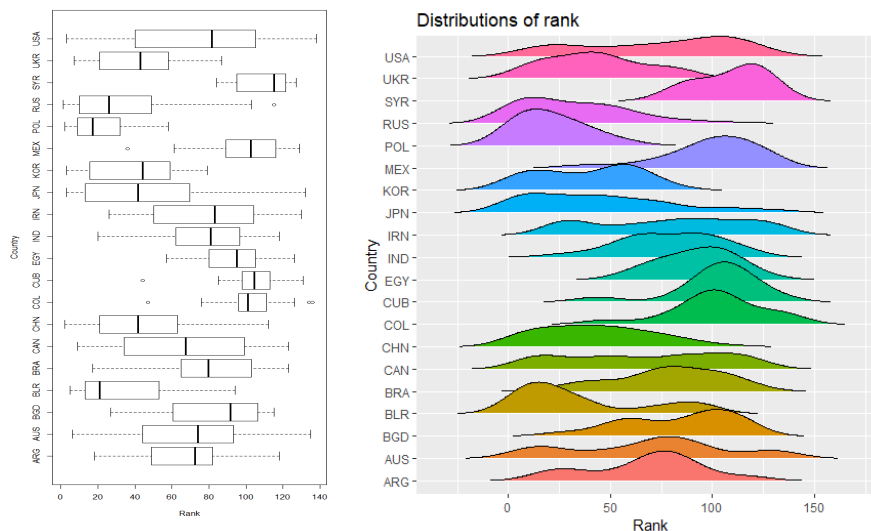


Figure 23: Distribution of rank for countries entered >10 times in the World Finals.

Interestingly enough, there are a lot of countries from local competitions that you do not see here. This is likely because winning teams are invited, but it is not mandatory to attend and compete in the World Finals.

In terms of (dis-)similarities on the distribution of solutions given in Appendix C, there are noteworthy similarities. The value in comparing them comes is that, although being influenced by a set of confounding variables (such as the problem difficulty and skill level of teams), the distribution of solutions over time explains something about the type of problems posed in a year. If this distribution is similar to that of the World Finals, this could mean that teams are more used to this type of problems and could perform better.

Looking at the figures, the overall difficulty of the World Finals is most comparable with the NEERC, NWERC and in lesser matter with the Latin American competition. There is similarity in terms of the overall number of solutions, the spread of solutions over time, diversity in problems solved and problem size, and popularity of the competition. However, not all countries participating in these local competitions have often participated in the World Finals. The Latin American competition’s distribution graphs are also similar in terms of solution spread and clustering but show relatively more easier problems.

Looking deeper at the influence of this similarity by considering the ranks of the teams of those competitions, the average rank of each year for each country is depicted below in Figure 24. As these are averages, when interpreting results, there must be kept in mind that sometimes an average is drawn over many teams (e.g. Mexican) and sometimes only one team per year (e.g. Turkey). Also, several countries have sparse data which influences the averages here (e.g. Turkey entering the competition in 2017 and Kazakhstan ending up 126th in 2018).

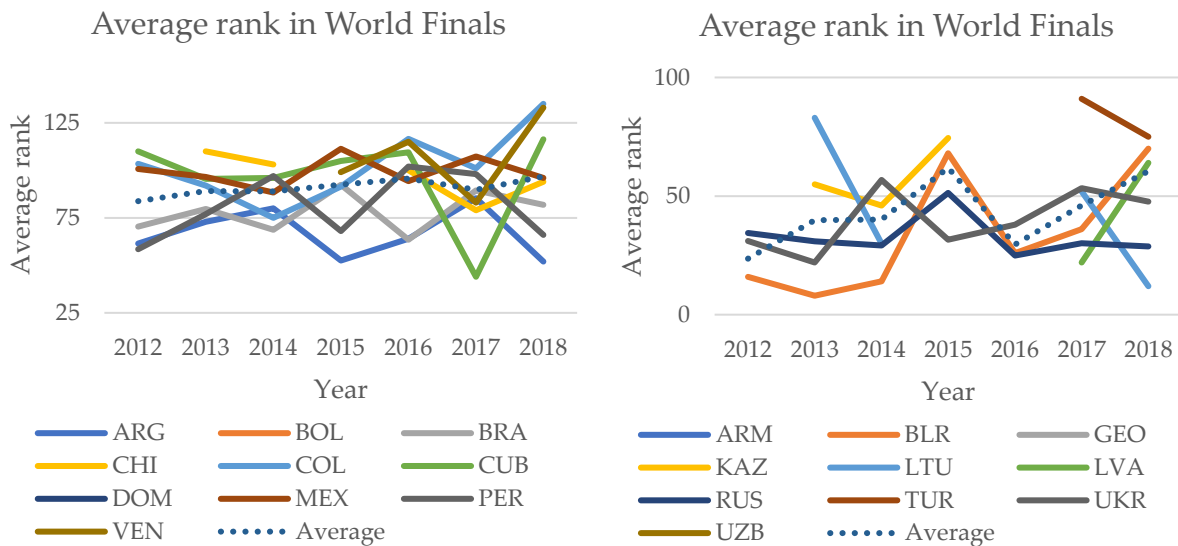


Figure 24: Rank of Latin American (left) and NEERC (right) countries in the World Finals.

Starting with Latin American, there could be seen that relative highest ranks were in 2012 and 2014 (on average). However, the solution distribution figures do not really show any much similarity for these years, nor do the solving ratios. This might have to do with the average rank being relatively low, so any effects possible influencing results in those years might not

really be visible. Looking at the NEERC, which had more similarity with the solution distributions, first there can be seen that the average ranks of these teams are way higher. In 2012 and 2016 countries performed particularly well and the solution distributions are similar for those years, as there are both more sparsely solved problems. It is not certain however if there is a causal relationship, as these years do not look distinctively different.

5.4 Performance of top teams

This section looks at the top performing teams to find out what makes them better than others. Several aspects of their game are considered to find out the distinctive traits that make them stand out among others.

To determine which part of the teams is considered the ‘top’, different aspects are considered. A top team must of course have a high rank and thus solve a large part of the problems (in this dataset, the top 5 solves almost three quarters of the problems (72.67%) on average, the top 10 solves 64.75% and the top 25 solves 52.07%), but also end up high in the competition. However, there are differences in the rewarding rules per competition, e.g. in the World Finals and Latin America, the top 12 teams are the medalists, and in other contests, only the top 3 are given prizes. It also differs which teams are eligible for advancement. As stated in the official ICPC rules “The highest-level regional contests advance teams to the ICPC World Finals”¹¹, but this does not specifically state any number of these teams, leaving the choice up till the regional contest organizers. As there is no consensus for the choice of a ‘top’ team, this analysis will look at the top 10 of teams, as this is a convention in sport and games, and is a good-sized portion of the general number of teams in each competition.

5.4.1 Origin of the top 10 teams

First the countries of origin will be looked at. In total, top10 teams come from 46 unique countries, and the most represented in terms of absolute numbers are given in Table 11. Aggregating the data gives a distorted image and shows only the biggest competitions (e.g. the absolute most represented country is the USA), so the data is split per region.

Europe		Latin America		World Finals		South Pacific		North America	
RUS	61	BRA	40	RUS	24	AUS	57	USA	634
POL	51	ARG	12	CHN	15	NZL	13	CAN	96
UKR	51	CUB	7	POL	7				
FRA	25	COL	5	USA	6				
GBR	25	PER	2	JPN	4				

Table 11: The top countries of origin from the top 10 teams.

The table shows that Russian and Polish teams do well in their local competition and also in the World Finals. For the South Pacific and North American region, there are only teams from two distinct countries participating, with the largest of the two also being the most represented as top10 team. Finally, Brazilian are the best performing in the Latin American competition. Beside the countries, the top 10 teams come from 271 distinct universities. As Table 3 in section

¹¹ <https://icpc.baylor.edu/regionals/rules>

4.2.2.1 showed, there are only relatively few universities which are represented often in general. The best performing universities amongst those are shown in Table 12 below.

Europe	Latin America	World Finals	South Pacific	North America					
University of Warsaw	26	Universidade Federal de Pernambuco	9	St. Petersburg National Research University of IT - Mechanics and Optics	11	University of New South Wales	11	University of Central Florida	39
Universitat Politècnica de Catalunya	18	Universidad de Buenos Aires - FCEN	8	Shanghai Jiao Tong University	6	University of Western Australia	9	Carnegie Mellon University	28
Taras Shevchenko Kiev National University	15	Universidade de São Paulo	6	Moscow State University	5	Monash University	7	University of Waterloo	27

Table 12: Top universities of the top 10 teams.

Especially for the World Finals and Latin American region, which are bigger competitions, there are no universities that always stand out, although some end up higher more often than others. In North America and Europe, the number of different universities is higher, but there are also some of them that significantly stand out more. Shown in Figure 25 below is that all competitions roughly follow the same distribution but differ only in size; there are always only a few universities that are performing very good (relatively), and more universities that are performing well but less often end up high.

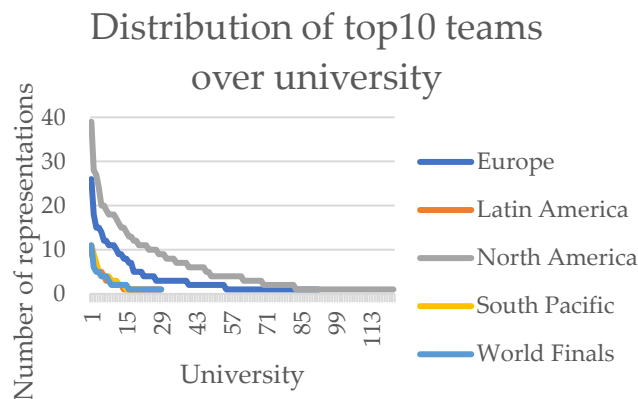


Figure 25: Most represented universities of top 10 teams.

5.4.2 Performance of the top 10 teams

First looking at attempts, top 10 teams use on average 1.943 attempts per problem and require 1.735 attempts to solve. This means that a top 10 team has around 200¹² minutes of penalty in their final time score, and an average solve ratio of 51.48% of the posed problems. Where the solve ratio is the number of attempts per problem divided by the number of correct solutions. In contrast, other teams have an average solve ratio of 39,302%, while using 2.544 attempts per problem and 2.009 attempts to solve. Interesting enough, the 1497 teams that did not solve anything did 10055 attempts on 444 problems, which is an average of 6.717 attempts per team,

¹² Average problems of 10.5736 x additional attempts (1.942 - 1) x penalty of 20 minutes = 199.207

and used on average 3.683 attempts per problem. This is higher than the other teams, likely because they got stuck.

For the all teams, the largest and most usual number of attempts taken is one, both to solve and as attempt. Using a second or third attempt is not uncommon. Higher relative differences between the total attempts and attempts to solve are mostly seen in the higher number of attempts. Also, there can be seen here that top 10 teams more commonly use one attempt.

Proportion of attempts per number of problems

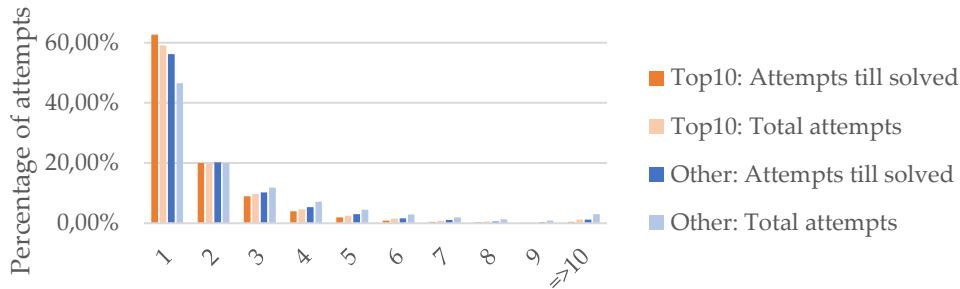


Figure 26: Proportion of attempts at number of solutions handed in.

Looking at the percentage of teams that solved a problem on first attempt, on first sight, a top10 team is more often able to solve a problem in one go. A normal team on average solves 56.22% of their attempted problems in one attempt, whereas a top10 solves 62.68% of the problems in one go. This means that overall, a top10 team is a bit more efficient. When looking deeper why, this is not solely because the top10 teams are consistently needing fewer attempts for each total number of problems solved, see Figure 27. The more likely reason is that they solve more problems (Figure 28), and when they solve more, they also do this more efficiently, as can be seen by the bars of the total solve sizes 13 till 10 existing only for top10 teams, and 9 & 8 being higher than the that of the other teams. Regarding the attempts and solutions, especially the distribution makes it apparent that the most important and basic difference is that top teams just solve more problems in general.

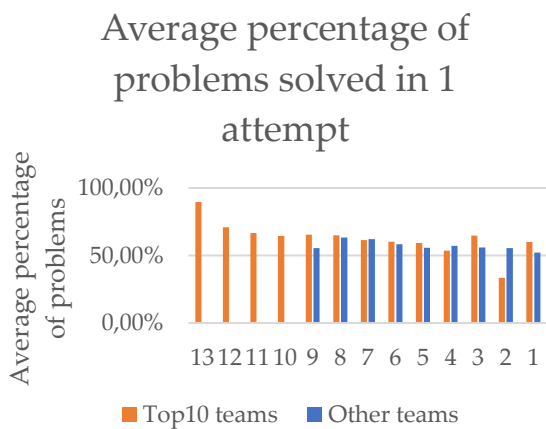


Figure 27: Problems solved in one attempt.

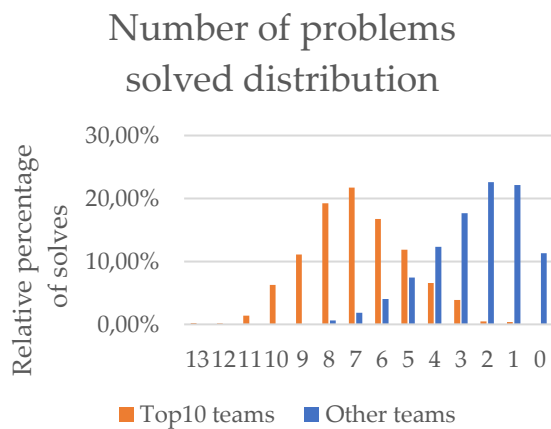


Figure 28: Distribution of problems solved.

There are however also other reasons the top10 is better. Top10 teams appear to solve a high proportion of the lesser solved problems. As can be seen in Figure 29a below, difficult problems are largely solved by top10 teams. Of the problems solved by 10% or less of the teams, an average proportion of 80.12% were top10 teams and even a proportion of 62.34% at problems solved by 30% or less of the teams. Note that this graph is ordered over the solving ratio and therefore looks only at attempts over solutions; it does not factor in that some problems with high solving percentage are only solved once or a few times. This can also be seen in the graph, as the problems with almost a 100% solving ratio are fully solved and attempted by top10 teams only. Furthermore, by default, top10 teams are a smaller group, which is why their proportion on the 'easier' problems is lower.

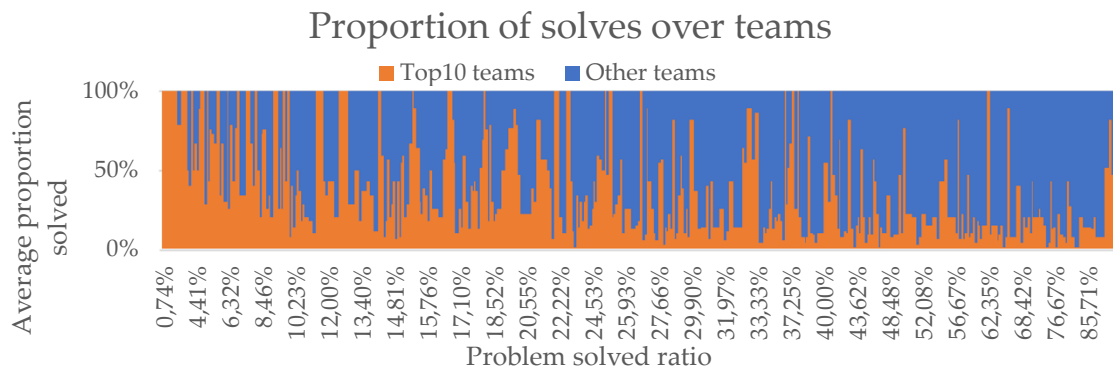


Figure 29a: Proportion of solves by two types of teams, order by problem solve ratio

Figure 29b is a different view of the data in Figure 29a, as it shows the relative proportion of top10 teams and other teams that solved each problem *over the teams* instead of over the problems (e.g. if 4 teams of the top10 solved a problem, their relative proportion is 40%). Here it becomes clearer that the less solved problems are solved in larger shares by the top10 teams. The problems with higher solve ratios (on the right side) also have an increasing higher proportion of the other teams solving them.

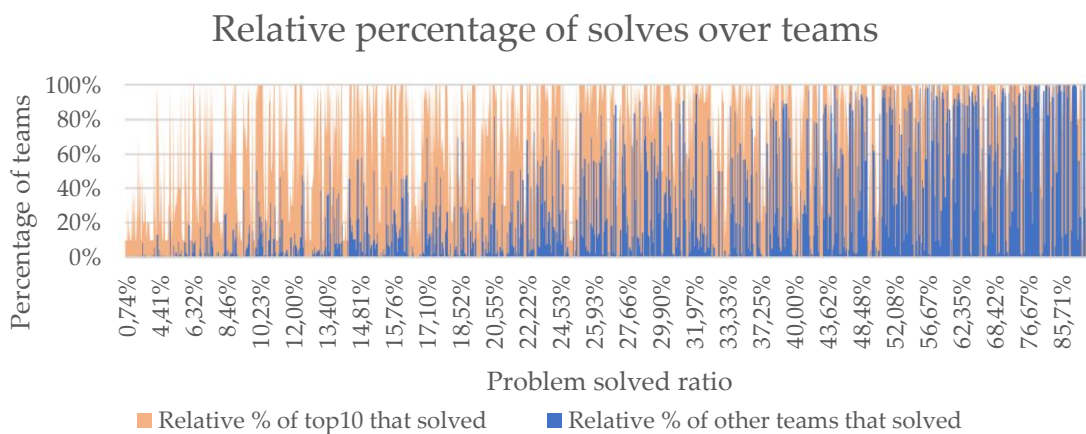


Figure 29b: Proportion of solves by two types of teams, order by problem solve ratio

In terms of uniqueness of the problems solved; there are 1199 problem solved in total and 1288 teams were the first to solve them (with some teams being equally fast, therefore the number

is higher than the total). Relative to this total, 1002 of the first solves (77.795%) were by teams in the top10, which is 1.486 problems on average solved first per top10 team. Moreover, 25.543% of the problems solved first were from the winning (nr1) teams in the dataset, with an average of 2.812 problems solved as the first. This means these teams are very often the first to solve one of the problems in the set.

Besides the attempts and solves, there is also the time. The average timepoint for solving a problem lays at the 153th minute, while for the top10 teams, this timepoint lays lower at the 119th minute. Teams not in the top10 even have their average at the 171th timepoint. This is according to expectations, as this means they solve problems earlier and hand in solutions at an average earlier timepoint. Figure 30 below illustrates this concept even further, where one can clearly see that most solutions handed in by the top10 are consistently at an earlier timepoint (on average) than those of other teams for almost every problem. However, for problems with a higher solving percentage, the two-point clouds become a bit more overlapping. Noteworthy is that some points lay above the 300-minute mark, because some competitions (e.g. the CERC) allowed for some extra time in certain years because of technical difficulties.

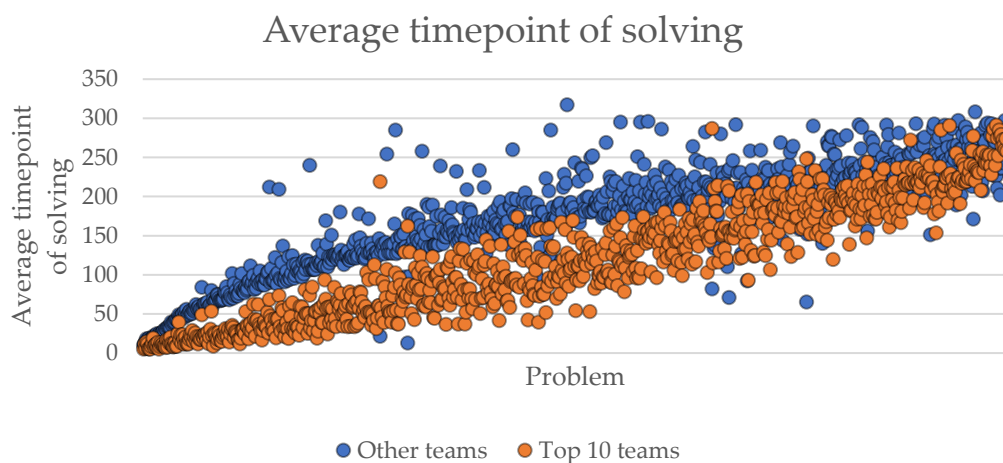


Figure 30: Average timepoint for solved problems, ordered by average timepoint

In addition to the timepoint of solving, there is also the actual time spend per problem. This is distinctly different from the time per problem, as it is the *interval* between the time a solution was marked as correct and the previous correctly handed in solution. Not that it is not entirely correct to assume that the entire interval is spend on one problem only, as teams could work individually on multiple problems in parallel, but it is assumed that the computer is used to program a single solution at a time. Below Figure 31a and 31b show these time intervals and that the approximated actual time spend per problem is lower for top10 teams and other teams separately. Two things are noticeable from the figures; the average time per problem (the polynomial lines) lays consistently lower for the top10 teams, and there are more dots in general for the top 10 team. Furthermore, these lines go up and down at the end, respectively. This is the case because there are some problems solely solved by top10 teams, for which they apparently took more time, and because there are some very easy problems, attempted only

once and were always solved. On average, a top10 teams takes 40.43 minutes per problem whereas other teams take 77.89 minutes. However, top 10 teams have a larger total time of on average 906 minutes for 6.882 solutions, while other teams require 344 minutes total for 2.462 solves (taking into account the number of problems in each competition). Of course, this total amount is higher because good teams solve more problems, but when considering the average number of solves as well, the top10 is a bit more efficient as well.

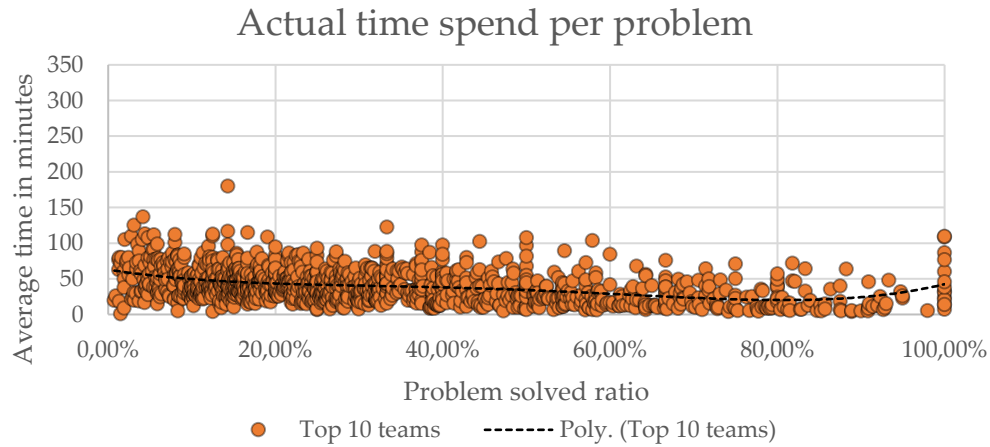


Figure 31a: Average actual time spend per problem, ordered by solving ratio

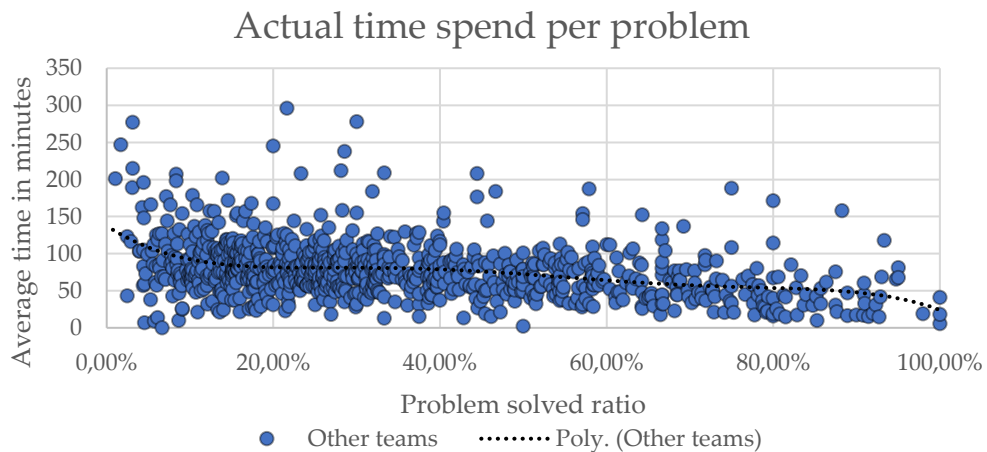


Figure 31b: Average actual time spend per problem, ordered by solving ratio

Chapter 6: Expanding data and analysis

After analysis of the data so far, this chapter focusses on expanding it and learning more. It does so by looking at problems and information extraction. As there is no meta-information readily available, this information must first be created. This chapter describes all efforts to extract information out of the problem descriptions and expand the database to analyze and combine it with the work done so far, gaining additional insights. The general structure of each section consists of the introduction of a topic with the reason it is written, the approach taken to get the data and ends with a description of the results. However, the descriptions of problems analyzed in the subsequent sections first need to be collected.

Collecting problem descriptions

The problems given to the contestants during the competition are handed out in the form of a single booklet, which may only be opened after the starting mark. Most of these booklets are stored on the websites of each respective competition in the form of pdfs. However, there is also an archive where all problems are collected and stored individually, called the ICPC Live Archive¹³. This archive is very suitable to link to the current dataset under analysis, as it further expands and completes the information about each competition.

With the scope of the analysis being confined to the more recent years (2012-2018) and specific regions, only a subset of the information available in the Live Archive needs to be extracted. Furthermore, the coding within the archive for the problems of the North American region is a bit inconsistent, as supposedly unique problem identifiers were reused between competitions and years for different problems. This makes it difficult to automatically process and link the files, which is why there is chosen to exclude the NA region at this point and only semi-manually collect data from Europe, Latin America, South Pacific and the World Finals; a total of 630 problems.

Still a significant number of documents needed to be collected, so manual extraction was no option. After the database structure of the website became apparent, specific competitions and years were located and retrieved. However, not all files within the scope were available. The archive is only updated up till 2017, so data from the year 2018 needed to be collected manually. This was done by visiting the website of each competition (sources in Appendix A.2) and downloading the problem set as pdf, after which individual problems were separated into single files.

6.1 Basic meta-information

In order to gain better insights on the variability in the data and to expand the data about problems, basic information out of the problem descriptions will be extracted. This data will later be used to expand on the patterns found and as input for further analysis.

First, the descriptions already contain an interesting piece of information; the time limit for which a given solution is allowed to run. This information can be easily extracted, while other simple info needs to be calculated. The extraction is based on counting different

¹³ <https://icpcarchive.ecs.baylor.edu/index.php>

types of elements in the text; there will be looked at the number of words, paragraphs, images, pages and numbers. The size and combination of these counts might also indicate something about the (perceived) difficulty of the problems. These simple statistics were generated by writing and running a text analysis script in python and running that on the descriptions. There is chosen to ignore the 'sample input' and 'sample output' sections, as these usually contain very contextual information (both in length as in content, which could be numbers or random text), which has no direct meaning in the counting of basic elements. Furthermore, the script also ignores counting repeating images (such as the logo of the competition, which is often given in the header).

Results

Running the script as described in previous paragraph resulted in an almost complete dataset containing basic information about each of the pdf, however for the CERC, NWERC, SEERC and Latin American problems of 2018, there is no time limit information available. It was capable of correctly extracting most basic information, after a non-trivial manual inspection on all files was done to remove footers, elaborate headers, page numbers, example in-/outputs and to solve minor incorrections. These calculated statistics are the most interesting when combined with the dataset from before. For example, the use of numbers in the problem description varies between competitions with the SWERC using the most numbers (39, std of 28.84) and the CERC the least (26, std of 16.23), which on itself this does not say very much. This combined analysis will be done later in section 6.3.

However, for other statistics, showing the distribution of the data and comparing between competitions could also indicate differences and/or similarities. Starting with the number of words and characters, which are (as expected) almost perfectly correlated and are the highest for the World Finals with an average of 419 words and 2096 characters per problem, whereas the lowest number is for the South Pacific, with on average 298 words. The standard deviations for these statistics of each competition are relatively high (average std of 135.82), so the differences are not directly informative. However, the number of words for the World Finals is significantly different from almost all other competitions (except NEERC and SWERC) under $\alpha=0.05$.

The number of pages per problem is relatively consistent with usually being two pages. Some competitions almost only use two pages (SWERC and the World Finals), while others tend to use only one (SEERC uses one page in 63.51% of the cases). The number of pages used rarely exceeds two, where the NEERC has the most problems with more pages (7 problems with 3 pages and 3 problems with 4 pages) although this is only in 11.90% of the cases.

Images or figures are sometimes used for explaining the problem and are usually dependent on the convention within a competition, with some competitions using almost none (SEERC) and others using one or more figures. When adding images, only including a single figure is most common (59.71%), although there are multiple occurrences of two figures used (26.21%), with the SWERC relatively using the most images. A note about the counting of images is that for some competitions (e.g. the South Pacific), images are also used for decorative purposes instead of purely being functional.

The time limit of a problem indicates how efficient a given solution must be for it to be accepted. Although very dependent on the topic and context, a lower time limit hint in the direction of both more efficient and thus harder, but also to easier and quicker solvable problems. The SEERC has the lowest average time limit of 3.9 seconds, while the South Pacific has the highest (8.73 seconds with a std of 17.44) because some problems have a very large time limit, e.g. there are two problems with time limits of 100 and 120 seconds. Time limit is not significantly correlated with any of the other information.

Finally, the number of the paragraphs per problem average around 14, but the standard deviations for each competition are high (average std of 8.36), making drawing significant conclusions difficult again. Also, there must be noted that the paragraphs here are defined and interpreted as the blocks of text with a whitespace between them which is dependent on the way the pdf is converted to plain text. This entails that the pdf's structure plays a role, and the extracted data shows that this sometimes results in minor deviations from the actual number of paragraphs. This resulted in paragraph information that is not completely reliable as automated generation of paragraph count is susceptible to perfect conversion of the structure, which is not always the case.

6.2 Topic detection

This section describes the different approaches taken to extract information from the problem description about its (algorithmic) topic and/or subject, with the final goal to link earlier patterns about difficulty and comparison with new information, in order to draw additional conclusions.

6.2.1 Gathering expert input

The first approach taken to get additional information about the problems is by consulting experts who are familiar with the ICPC and are capable of analyzing problems. The way these experts are consulted is by inviting them to use a specially created online platform, where there can be navigated through the problems from the chosen regions (including the World Finals) from 2012 till 2018. The descriptive text of each those problems are gathered from the ICPC Live Archive in the form of pdfs and shown on the platform. As described before, gathering these documents was a tedious task, as there are 1342 distinct problems in the database. After the platform was created and hosted online, experts were invited to use it. An included feature is the option to indicate if a problem has been seen before, making submissions more reliable as this likely means they have created or solved that problem in the past. After a certain problem is analyzed multiple times, it is removed from the list, such that both the data can be assured to be reliable, but input for the most different problems possible is received.

In order to gain the desired data and get as much use out of the experts as possible, the experts are asked to fill in multiple questions on how they perceive a problem in terms of its category, and also in terms of its difficulty. This difficulty is subdivided into the overall difficulty, the solution difficulty, the in- and output-difficulty and the coding amount required. Both measures combined will give a better view on the difficulty of each problem. Figure 32 shows how the added information extends the data structure shown before in Figure 6, section 3.2.

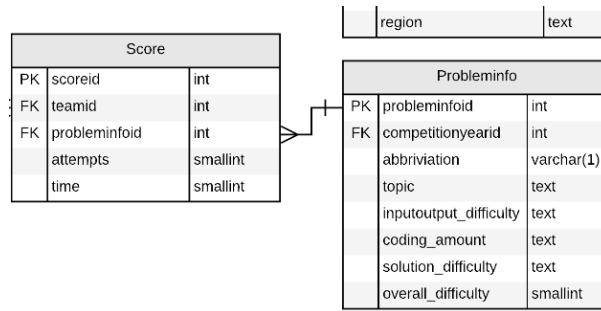


Figure 32: Added attributes to the *Probleminfo* table in the data structure.

The added attributes to the *Probleminfo* table will contain information about the problem topic(s) and difficulty, broken down into several factors. These are the amount of coding it requires (minimum, medium, large), the difficulty to identify the proper algorithmic solution (easy, medium, hard), the difficulty of manipulating the input and output (easy, medium, hard) and the overall difficulty of a problem ranked one to five, with one being very easy and five being very hard.

Results

The submission platform was programmed as a website with a simple interface where the experts could quickly navigate problems. To provide a smooth and fast experience, some meta-data was automatically extracted and filled into the form. As seen on the right side of Figure 33, with only a few fields to be filled in, data can be submitted fast. On the left, each year and competition can be navigated until a problem is chosen; then the pdf version of that problem is shown. Beside ensuring data quality with some basic security such as logging in, input checking and submission logging, the website has several features. It includes an information page with the research goal and an acknowledgement page where all experts that provided input were mentioned, ordered by an updated number of submissions.

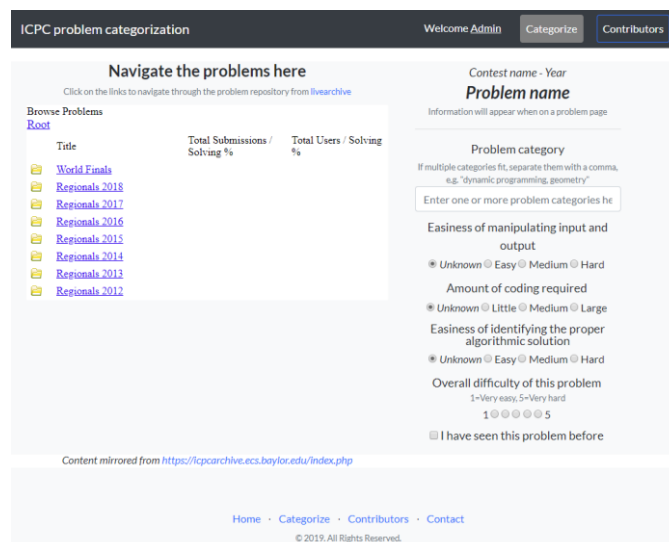


Figure 33: Submission page.

However, despite an uptime of over two months and multiple efforts to call for input among experts, not many submissions came in. In total, there were 99 submissions received from 24

different experts, with some experts providing many inputs (one expert provided 24 submissions) and most experts providing only a single submission. Received submissions are on 57 unique problems, mostly from the Latin American region (87 of the submissions). Most expert indicated they had seen the problem they submitted information about before (81/99). All received submissions were manually processed, cleaned and combined when there were multiple inputs. The topics provided were semantically combined by focusing on the most occurring category, leaving only one or two categories per problem. Where necessary, too specific topics were generalized into a broader subject, for example a submission such as “Dijkstra” is a specific type of graph theory. This resulted in nine different topics; Data Structures, Dynamic Programming, Geometry, Graphs, Greedy, Math, Number Theory, String Manipulation and Ad Hoc. At this point, the difficulties of each problem submitted by the experts were not processed and analyzed due to insufficient information being available.

The low amount of submissions, which is almost exclusively for a single region, makes it very hard to generalize and analyze the data on its own. Therefore, the size of the set is insufficient to use at this point of time; it cannot directly be used for analysis. Cleaned and raw submissions of the experts are attached in accompanying CSV files.

6.2.2 Topic extraction with machine learning

With efforts in the previous section yielding insufficient results, there will be looked at analyzing the problem files algorithmically and automatically to gain additional meta data on the topic of each problem. Two different approaches will be used; building and optimizing an unsupervised LDA model and building and training supervised models using an additional dataset.

6.2.2.1 Topic detection using unsupervised LDA

The topic of a text is something that can often be fairly easily determined by a human reader, as the semantic meaning and relation between words are known. By combing the meaning of words, prior information related to the topic the text is about and looking at the appearance and frequency of specific words, a subject can be found. Machine learning techniques for text categorization deploy techniques similar to the latter approach and can learn to categorize a text with high accuracy. Most of these techniques require a specification of all possible topics upfront and often need to be trained on labeled data. However, with the task at hand, the problems underlying algorithmic topics are not known up front. Clustering similar descriptions of each problem and looking at words that are specific for each cluster is a way to discover similarities between them and to find possible distinctive words that together can be interpreted as a topic. One of the techniques that is suitable for this task is Latent Dirichlet Allocation (Blei, Ng & Jordan, 2003), which is an unsupervised learning method. It builds a probabilistic model for interpreting textual documents as a mix of several topics, where a topic is seen as a multinomial distribution of words. The method learns the distributions of words for each of a predefined number of topics, where it sees a document as a text that consists out of a mix of topics, where the words determine to what degree each topic is represented in a text.

Using existing implementations of LDA, the algorithm will be applied to the problem description dataset with the goal of discovering coherent topics. However, first some preprocessing of the data must be done as according to Denny & Spirling (2018), the way data is preprocessed for unsupervised methods can have large impact on the results. Preprocessing is often the first step in most research involving text analysis, and researchers have employed multiple different ways of doing so. One example out of many papers is that of Abualigah & Khader (2017), who used tokenization, stop word removal, stemming and finally term weighting (calculating term-document frequency) to represent their text data in numerical form. This numerical form is often called a Bag-of-Words representation. Denny & Spirling (2018) extend the list of possibilities with lowercasing, in- or excluding special characters and numbers, n-gram inclusion and removing infrequently used terms. Furthermore, they show that notable papers in the field of unsupervised learning show no consensus in using a specific set of preprocessing actions out of these techniques. Therefore, multiple differently pre-processed datasets will be used as input for the LDA, evaluating the resulting models to see which works best.

Model building

With multiple different setups being used in the field, models with differently preprocessed data will be built and compared below. R and the implementation of LDA in the R-Package 'textmineR'¹⁴ are used to build and visualize models. The models are compared on the average probabilistic coherence of all topics, where coherence is used to evaluate how good a topic is based on the top words ($M=5$ in this section). The goal of this section is to retrieve meaningful word sets, which is exactly what this measure has proven to do (Rosner, Hinneburg, Röder, Nettling & Both, 2014), and finally classify these sets into topics.

After some initial experimenting it became clear that, although the flow of adding/removing pre-processing steps seems straightforward, there is an interplay between the different steps taken to clean the data. Furthermore, most models identified a number of topics that is much higher than you would reasonably expect based on the number of topics provided by expert input. As the experts manually identified only nine topics, the search space for the k number of topics should be limited to be not much higher in order to ensure meaningful subjects. To better suit this context, a more semantically influenced approach of selecting preprocessing steps is necessary. The reasoning for each step, processing text into a cleaner format, is explained below:

First, including n-grams could be beneficial for the models, as the removal of (in-) frequent terms automatically handles the in- or exclusion of these n-grams. If the n-grams have high cohesive power and are thus useful, the model will have the option to include them, so n-grams must be added.

Beside non-useful n-grams, a problem's description logically contains a lot of unnecessary words, with most of these words being stop words. They will be removed, also because that typically leads to better results in text analysis (Schofield, Magnusson & Mimno, 2017). Also, (in-)frequent terms that appear only in less than one percent- and frequent terms

¹⁴ <https://CRAN.R-project.org/package=textmineR>

that appear in almost all documents were removed, as these are unlikely to be discriminating (Grimmer & Steward, 2013) and speed up the analysis of the corpus (Denny & Spirling, 2018) as it is shrunk down a lot. Preliminary experiments showed that removing frequent words beyond the 99%-mark as proposed by Denny & Spirling could have a positive impact on the results, so experiments considering different cutoffs will be done.

The occurrence of special characters (e.g. <, =, but also one-character abbreviations) can possibly be correlated with certain subjects such as math or geometry, so these must be kept. However, punctuation is used for sentences and therefore is likely to be noisy and overly frequent, so dots will be removed. Additionally, numbers could also be correlated but are also expected to be very noisy, as the text could for example contain tables. Preliminary tests show that removing the numbers leads to better results, so these are kept out of further models.

As the words that starts each sentence in the problem description, or other capital words, have no specific semantic meaning in this context, lowercasing all words represents the descriptions better. In addition, stemming could also be performed, of which the impact stemming is unsure, so there must be experimented with this technique. Porter's word stemming algorithm, as described in (Willet, 2006) will be used during these experiments.

Making LDA models with above mentioned parameter settings results in Figure 34 below. Results of only the best models are shown here (as the first few models performed relatively worse), where simultaneously the number of topics (hyperparameter k) is tuned. The red color means frequent terms up till 99% are kept and blue means up till 50%. Striped lines show the models with stemming and solid lines show the models without. Finally, the black line represents the performance of an LDA model with up till 90% frequent terms kept, which uses a corpus which has the same results both with and without stemming. The best model is the one with the highest average coherence.

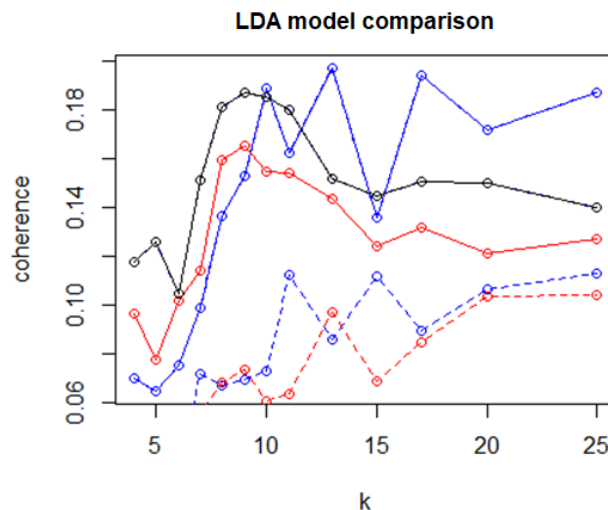


Figure 34: Tuning differently preprocessed LDA models on k.

Results

As can be seen in Figure 34, the best performance (and an overall R² of 11.47%) is by the model with no stemming, keeping terms that are <50% frequent (1843 out of 239628 using 4-grams), at a k-value of 13. This means that there are 13 potential topics identified, with an average

coherence of 0.1970. Table 13 shows the top 5 words, coherence and prevalence for each topic. Distilling meaningful subjects seems to be possible for some of these topics, but not all identified topics and terms seem equally useful for identifying the underlying programming problem. Chuang, Manning & Heer (2012) describe another measure for ranking and filtering terms called salience, which is a measure of distinctiveness of words. They claim that it can aid rapid classification, lead to disambiguation of topics and helps to identify potential bogus topics. This measure has been implemented by Sievert & Shirley (2014), who also define a measure of relevance to further improve the selection of salient terms, as they claim that a relevance term of lower than 1 (the default and maximum value is 1) can improve topic interpretability. Changing the setting for this measure results in a different order of the top words. These measures, which are implemented in the same authors' R package 'LDAvis'¹⁵, will be run on the current problem dataset. Sievert & Shirley also implemented a way of visualization these topic distances with an inter-topic distance map (Figure 35), showing how different the set of words are for every topic. All information and the interactive visualization will be shown to an expert, who will then be asked to label the most likely topic of each group of words. The outcome of this labeling is also shown in Table 13.

Topic	Coherence (M=5)	Prevalence	Top terms	Likely topic
1	0.21131	12.084	sequence,numbers, cases_line, testcases_line, file_tescases_line	Number theory
2	0.09045	9.648	time, water, cost, path, distance	Stories about transport: <i>Topic unknown</i>
3	0.22616	9.116	string, word, letters, characters, words	String manipulation
4	0.11105	8.993	figure, polygon, display, consists, sample	2D Geometry
5	0.31958	8.052	cells, cell, grid, left, row	Solving puzzles: Combinatorics
6	0.12259	8.162	order, total, description, follow, testcase_output	Graph theory
7	0.16290	7.792	point, points, coordinates, area, xi	3D Geometry
8	0.02653	8.218	consists, program, single, separated, numbers	Ad hoc
9	0.21227	6.912	representing, indicating, line_integer, representing_number, minimum	<i>Topic unknown</i>
10	0.16280	5.801	game, player, room, card, players	Stories about games: Combinatorics
11	0.16808	6.052	city, ai, cities, road, bi	Location math, sets/collections
12	0.41576	4.513	tree, graph, vertices, vertex, edge	Trees, graphs, data structure
13	0.33220	4.657	print, inputline, number_testcases, number_test, follow	Ad hoc

Table 13: Statistics for each identified topic by the best LDA model

¹⁵ <https://cran.r-project.org/web/packages/LDAvis/>

Both the top terms and the salient terms show meaningful differences between the sets of terms. The inter-topic distance (Figure 35) also shows these visually, where for example the topics on the right side of the quadrant are more similar in terms of mentioning the output/input in the description, and very different word sets are placed further away from the others, indicating potentially completely different topics. Showing all information combined made it possible for the domain expert to identify most of the subjects of each topic. Interesting enough, some topics clearly come up and were very easy to identify, while others could not be distilled at all. Based on the reasoning of the expert and the given labels, it seems that there are five types of problem descriptions (in this dataset):

- Descriptions with an obvious topic such as string manipulation and geometry problems. There are key distinctive words associated with these problems, making them easier to identify.
- Descriptions which could hint towards one of multiple topics that fall within the same kind of category, like topic 12. It is completely dependent on the context if such a problem e.g. focusses more on building the structure of a give object (such as a tree or graph) or more on the search through it, but distinctive words for the category are used so a human reader can more easily identify these type of topics.
- Descriptions given as a story, in which the algorithmic solution is not always apparent and masked partially within the story. Examples are topic 2, 10 and 11. The likely algorithmic topic cannot always be clearly distilled based just on the words here, more context information and knowledge about algorithms is necessary to identify any topic.
- Descriptions seen by the classification as separate topics, which are just different ways of describing the same type of problems, e.g. topic 8 and 13 are both about the general ad hoc problems, which can apparently be described in at least two different ways.
- Descriptions with no or not enough distinctive words to be identified as a separate topic. Looking at the manual input from the experts before, topics such as brute force and greedy algorithms cannot be found using the build model here, or by only looking at groups of words occurring together.



Figure 35: Inter-topic distance map

Conclusion

So, apparently the model is identifying something other than just the algorithmic subject of the problem; it identifies the characteristics of the story that is written about it. This likely has to do with the way the descriptions are written, which is also dependent on the topic, e.g. problems about graphs and trees appear to often use these exact words 'graph' and 'tree' in their description and are therefore more identifiable, while other type of problems are more hidden in a story. As the expert put it: *"There is no way to say that a problem is, for instance, about dynamic programming by just looking at these words. It is even a general saying that if you don't know how to solve a problem, you should likely use dynamic programming"*. This indicates that it is not possible to find all algorithmic topics in the text using the current unsupervised LDA technique; only some topics can be identified with more certainty as these use more distinctive words. This is something that the teams can keep in mind when tackling and choosing which problems to solve.

6.2.2.2 Topic detection using supervised algorithms

Another approach is to make the problem of detecting supervised, by letting an algorithm learn on a labeled dataset first after which the topics of the problems at hand can be tagged. This inherently uses the set of topics of the dataset chosen but makes it possible to validate and iteratively improve the model to see how well it generalizes. The goal of this section is to build a model that can identify topics in problem text as accurately as possible and run that on the problems of the ICPC.

Data collection and processing

There is a need for a significantly sized tagged problem set in order to properly train a supervised model. One platform that contains such a set is CodeForces¹⁶, a social network for programming enthusiasts where online programming contests are held. Through their API, meta information about problems can be retrieved, such as a problems' identifier, tags and difficulty. The process of collecting and cleaning the data was as follows:

First, all meta-data was collected, which consists of 5166 problems of which 4993 are tagged by at least one tag. Only the problems that are tagged are of importance and only the problems' name and tag fields will be used, so only these rows and columns are kept. Processing this dataset however has two potential issues: there are as much as 36 different topics with some being represented by little problems, and up till eleven different tags are given to each problem. This high dimensionality makes model training and interpretation harder and more time consuming. Also, the set differs much from the ICPC dataset, making comparison and fitting more difficult. To tackle these potential issues, tags that occur the most infrequently, in only ~2% of the problems or less than 100 times, are removed (thus also removing problems containing only that tag). Problems with six or more tag occur infrequently (65 problems), so they also can be left out of scope. Finally, there are some problems that are tagged as '*special' of which the meaning cannot be found, so these are also removed. This results in a scoped dataset of 4665 problems, maximally tagged by five different

¹⁶ Description: <https://codeforces.com/help>, Dataset: <http://codeforces.com/api/problemset/problems>

topics, out of a total of 23 different topics. These topics are: *binary search*, *bitmasks*, *brute force*, *combinatorics*, *constructive algorithms*, *data structures*, *dfs and similar*, *divide and conquer*, *dynamic programming (dp)*, *disjoint set union (dsu)*, *geometry*, *graphs*, *greedy*, *hashing*, *implementation*, *math*, *number theory*, *probabilities*, *shortest paths*, *sortings*, *strings*, *trees* and *two pointers*. However, this dataset does not yet contain the raw text of the problems, so these must be retrieved in a different way.

In contrast to the ICPC problem pdfs collected before, all problems of the CodeForces website have already been scraped before and are stored in an online repository (Fadel, 2018), containing 5192 problem pdfs. Although this repository does not contain all problems up till the current date, the size of these problems is sufficient enough. This set will be used to speed up the process.

After collection of all pdfs, each file was subsequently converted into text to be used as input. This was a non-trivial task for which the previously mentioned script in Section 5.1 was adapted and used. Again, the example input and output sections were left out of each file. There must be noted that, although almost all resulting files are properly structured, a manual inspection found some files having their sections rearranged and sentences spread over multiple lines. However, as files will be converted into a bag of words representation later, this will have no impact on the analysis. Matching these problems on the meta-data collected before resulted in a final dataset of 3830 tagged problem descriptions. Most of the problems have been tagged with up till three topics (roughly 85%). There are also some moderate positive correlations between the labels, such as *trees* and *dfs and similar* (0.361), *graphs* and *dfs and similar* (0.419), and *shortest paths* and *graphs* (0.351). These correlations are understandable as the topics are also related in real life. Noteworthy is that the spread of topics over all problems is imbalanced. Some topics such as *implementation* are abundantly present (33% of the problems) while others such as *shortest paths* only appear 96 times, which is only in ~2.5%. The topics less represented might be more difficult to be learned, although it is likely that not all topics can be equally well distilled from the text anyway.

Model tuning

Similar work has been done by Bora & Sinha (n.d.), who also looked at predicting a single algorithmic tag for programming problems. They state that the actual underlying subject of each problem is masked in the deeper meaning of the description, which is something that might indeed hinder the analysis. Even worse, they are not capable of achieving accuracy marginally above the majority class. However, this section will use different algorithms and build models on all tags, taking on a different approach that might give other and better results. The approach used is known as multiclass-multilabel classification, where the goal is to assign one or more labels out of k possible classes to each item (Dekel & Shamir, 2010). There are multiple ways of handling such a problem that all come down to two high level approaches; problem transformation and algorithm adaptation (Tsoumakas & Katakis, 2007). Both ways are tried, where the focus lays on achieving both high recall and precision. Notable is that accuracy is a less appropriate metric to evaluate on given that 66% or more of the data for any topic is not applicable (e.g. always 'not' predicting a topic already gives 66% or higher accuracy per topic), so F1 score and categorical accuracy are used as evaluation metrics.

Multiple approaches are tried here, and models are trained on an 80% random sample, evaluated on an 10% sample and tested on the other 10%. Python was used to write, run and evaluate models, were several implementations and established approaches from papers were tried. Unless stated, no specific preprocessing was performed, because Bora & Sinha (n.d.) found in their literature survey that performing preprocessing such as removing stop-words and lemmatizing results in a worse performance in this specific context. The data is set to include up till trigrams and a maximum of 5000 features. Below section describes the set-up and efforts to optimize each model.

Problem transformation approaches

A common way to transform the multilabel multiclass problem is to transform it into one or more simple-label classification tasks, of which binary classification is the most widely used (Katakis, Tsoumakas & Vlahavas, 2008). This type of classification predicts the probabilistic percentage for each individual class separately to see if it does or does not belong to a topic, where predicting each topic is seen as an unconnected task. For this type of approaches, the data was shaped into a TF-IDF matrix. The first model was made using binary classification applying logistic regression, which has proven to yield relatively high performance in a textual context (Zhang & Oles, 2001). The tunable hyperparameter for logistic regression is the regularization term C, which determines how complex and tight the fit might be with respect to the data. Tuning this parameter on the evaluation set and calculating the evaluation metrics at a cutoff of 0.5, results at the highest F1-score at C=20, as can be seen in Figure 36. The next step is to determine the most optimal cutoff point at which a probability for a given class is rounded down or up in order to maximize the F1-score, as there is a tradeoff between precision and recall (Buckland & Grey, 1994). As there is dealt with multiple classes, these metrics represent weighted averages. Different splitting values are shown in Figure 37, where the highest F1-score is at a cutoff of 0.16. A logistic regression model is built with these best settings achieving a weighted average F1-score of 0.41. The evaluation metrics are split on topic and shown in Appendix D, indicating that this model is capable of identifying only some of the topics with a reasonable level of accuracy. For example, it is relatively good at identifying problems that about *trees* or *probabilities* but cannot detect problems about *bitmasks* or accurately find *two pointers*-problems.

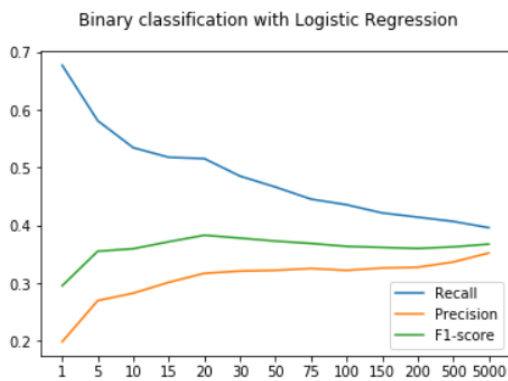


Figure 36: Logistic model; tuning on C.

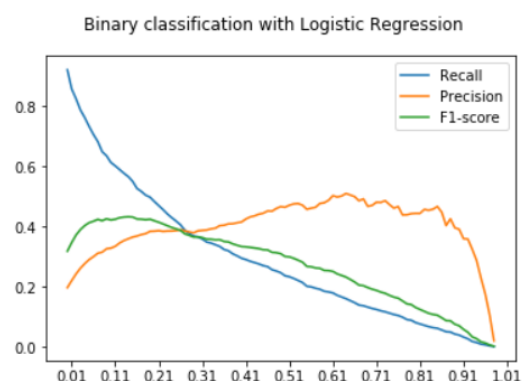


Figure 37: Logistic model; determining cutoff.

Next, some experiments with Multinomial Bayes were performed, as this is a fast and understandable classifier. However, confirming the Godbole & Sarawagi (2004), these models are fast but result in only modest performance with a F-score of 0.26. Results per topic for this model are shown in Appendix D.

Another approach is tried inspired by Godbole & Sarawagi (2004), who mention and show in their paper that Support Vector Machines are accurate and relatively good performing classifiers. An even stronger conclusion is drawn in the research from Santos, Canuto & Neto (2011), who state that SVM had the best results among the problem transformation methods on all their datasets. This method is implemented and tuned on the current dataset via a 2-fold cross validated grid search for the parameters (best values are $C=2000$, $\gamma=2$ and 0.001) with a radial basis function kernel. These models have a F1-score of respectively 0.31 and 0.18, of which the per-topic evaluation is again shown in Appendix D, in Figure D3 and D4. Both models are kept as they perform differently per topic, where it is clear that almost half of the topics cannot be identified by either model, but are reasonably good at identifying the others.

Algorithm adaptation methods

Beside the problem transformation methods, the other approach is adapting known algorithms to work in a multilabel context. For this type of methods, the data was tokenized, converted to sequences and padded. One of such approaches is described by Zhang & Zhou (2007), who build a multilabel version of the K-Nearest Neighbor algorithm. They show that it outperforms some other well-known algorithms on multiple datasets. K-NN has a tunable parameter k , which is the number of neighbors that should be considered when classifying new instances. Below Figure 38 shows that the weighted F1-score is the highest at $k=16$ on the evaluation set, so this setting is used to predict the test set. Results of these predictions are a weighted F1-score of 0.18, and topic specific results are shown in Appendix D.

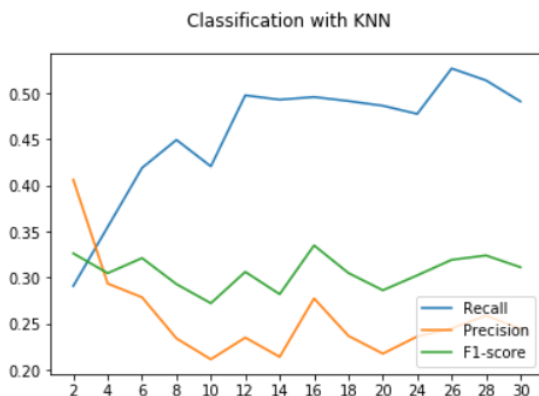


Figure 38: K-NN; tuning on k.

Layer (type)	Output Shape	Param #
input_216 (InputLayer)	(None, 600)	0
embedding_202 (Embedding)	(None, 600, 100)	500000
global_average_pooling1d_20	(None, 100)	0
dense_95 (Dense)	(None, 23)	2323
Total params: 502,323		
Trainable params: 502,323		
Non-trainable params: 0		

Figure 39: Best NNET model layers.

Another type of algorithms are neural networks, known for being able to capture non-linear and complex relationships between variables. One of such types has been used and researched a lot lately, is called a convolutional neural network (CNN), which can achieve good results with only little hyperparameter (Kim, 2014). Although being relatively slow to train and test, it achieves good performance in practice (Joulin, Grave, Bojanowski & Mikolov, 2016). Several different implementations of NNET and CNNs were tried here, where there was experimented

with different inputs, hyperparameter settings and combinations of layers (e.g. adding bi-directional layers, having multiple inputs such as LDA and using other algorithm's predictions as input). There was tuned using categorical accuracy as metric, but no combination of explored settings would yield a validated categorical accuracy higher than 0.25. In the end, the implementation of a CNN using Keras (Chollet, 2018) was used and adapted for this context, excluding the convolutional neural network layer, with an embedding layer using 100 embedding dimensions and a maximum input length of 600. These numbers were found using a grid search with the final settings shown in Figure 39, leaving just the number of epochs to be optimized. Figure 40 shows the training and validation accuracy for each epoch, where the best results are gained at 13 epochs. Tuning the optimal cutoff for this model can be seen in Figure 41 where the best value is at 0.12; values equal or above this value are classified as 1 (= the class) and values below as 0. The final F1-score for this model on the test set is 0.43. Topic specific results are in Appendix D, Figure D5.

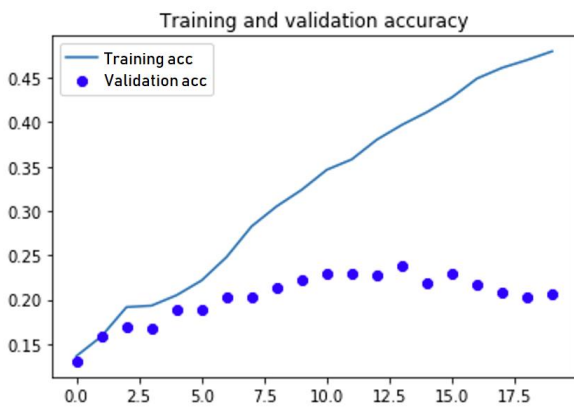


Figure 40: Convolutional neural network; tuning on epoch.

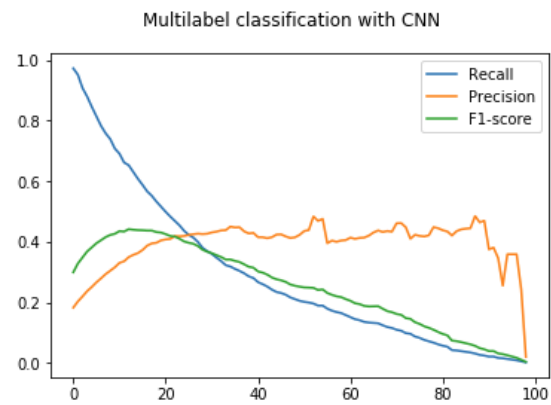


Figure 41: Convolutional neural network; determining cutoff.

Final model building

The overall best performing models at this point are the neural network and logistic regression models, both being better at certain topics. However, these models are still not accurate enough to be generalized, although multiple different setups are tried. As the models themselves seem unable to be improved upon, the only other factors that can be influenced are the data itself and the choice of algorithm per topic. Furthermore, speaking with a domain expert hinted in the direction of looking at in- or excluding words of certain length, because that might leave more specific words for certain topics. Against the findings of Bora & Sinha (n.d.), preliminary experiments with the preprocessing data, excluding dots, control characters and words of certain length improved results. Further extensive experiments with preprocessing setups were done using the logistic regression model, as this is a more understandable and less time-consuming model to build. Extensive grid search for the best data setup for each topic was performed. Tuning on the maximum number of features [5000,8000,10000], the exclusion of words with length up till six characters and the inclusion of n-grams up till size five, a set of 1923 models (84 for each topic) was build and tuned on the evaluation set. This resulted in the best data setup settings specific to each topic. Using these setups, a separate model was built using the best NNET, Logistic Regression, MNB and SVM

models of previous section and used to predict each topic, where getting the highest possible F1-score while maximizing the precision was the goal (e.g. the results of the generalizable model should be precise, as this leads to accurate data where it is of lesser concern if possible fitting topics are missed, as the provided data is at least correct). The choice of algorithm and preprocessing per topic are shown in Appendix D, Table D1. This resulted in a model with an average weighted F1-score of 0.31, but with the highest precision for each class out of all binary classifier models. Topic specific measure scores are shown in Appendix D, Figure D6.

Results

The best model built on both the evaluation and training set, using a specific algorithm for each topic, is the most capable but cannot detect all topics (accurately enough). For this reason, topics that have less than 0.5 accuracy are not generalized, leaving 17 topics shown with their scores in Figure 42. This model is reasonably accurate for most topics but does not have high recall values. With the evaluation scores of the final model in mind, there must be noted that further analysis using this model will be influenced by its lower F1-score, and generated information by this model might not be fully accurate.

	precision	recall	f1-score	support
combinatorics	0.83	0.22	0.34	23
constructive algorithms	0.75	0.14	0.24	43
data structures	0.66	0.28	0.40	67
dfs and similar	1.00	0.09	0.17	33
dp	0.76	0.15	0.25	85
dsu	1.00	0.11	0.20	18
geometry	1.00	0.24	0.38	21
graphs	0.73	0.34	0.47	32
hashing	1.00	0.33	0.50	6
implementation	0.55	0.39	0.46	120
math	0.53	0.27	0.36	73
number theory	0.73	0.31	0.43	26
probabilities	1.00	0.27	0.43	11
shortest paths	1.00	0.10	0.18	10
sortings	1.00	0.06	0.12	47
strings	0.52	0.65	0.58	23
trees	0.90	0.31	0.46	29
micro avg	0.64	0.26	0.37	667
macro avg	0.82	0.25	0.35	667
weighted avg	0.74	0.26	0.35	667
samples avg	0.33	0.25	0.27	667

Figure 42: Final model evaluation metrics.

This final model is run on the ICPC dataset to predict its labels. However, of the 630 problems, only 225 can be tagged. This is according to expectations, as the model’s recall is relatively low. Even more, there must be kept in mind that these specific ICPC problems are only for the western competitions, that could also actually be about fewer topics or be written in a manner different than the training set, although the model itself likely plays the biggest role. Table 14 shows the number of tags for each topic, where can be seen that some topics such as *implementation* are way more prevalent than others. All problems in this set have a single tag (225), 41 have two-, four have three tags and one problem has four.

Topic	Number of tags	Topic	Number of tags
Combinatorics	2	Math	28
Constructive algorithms	4	Number theory	5
Data structures	10	Probabilities	1
DFS and similar	2	Shortest paths	1
DP	3	Sortings	0
DSU	3	Strings	22
Geometry	6	Trees	15
Graphs	32	Binary search	3
Hashing	0	Brute force	46
Implementation	88	Total:	271

Table 14: Number of tagged documents per topic.

6.3 Analysis with generated data

This section looks at the newly generated data on both the topic and basic meta-information of each problem created in previous section. Although the generated data is limited and only seven topics have been identified ten or more times, tags are fairly equally spread-out over all regions and years in the given scope. The problems that are labeled, focusing on more often identified topics (≥ 10) will be analyzed in below section.

Topics and difficulty

If all problems were labeled for an entire year, competitions and competition years could be compared, where trends could possibly be spotted about which competitions and years have focused on which topics. For instance, the World Finals could mostly have problems about *graphs*. However, the size of the data is too limited to draw conclusions on (e.g. Northwestern Europe had 4 problems about implementation, whereas Northeastern Europe had 16, but there is too much missing data to say anything about this). There can however be looked at the correlations between topics and other statistics from problem descriptions, to see differences.

Topic	Average Solving%	Std	Average time	Std	Actual time spend	Std
<i>Data structures</i>	7.486	10.340	205.209	69.424	68.522	43.786
<i>Graphs</i>	25.473	18.025	167.721	77.857	59.246	44.909
<i>Implementation</i>	34.263	25.268	99.918	82.563	52.169	82.563
<i>Math</i>	30.197	28.944	89.448	83.433	48.909	51.440
<i>Strings</i>	30.015	17.585	114.604	76.035	59.684	52.678
<i>Trees</i>	23.044	21.207	128.024	90.722	55.554	43.535
<i>Brute force</i>	25.950	23.243	115.792	79.266	61.349	54.477

Table 15: Difficulty metrics per topic.

As can be seen in Table 15 above, there are differences between topics, although standard deviation for most topic's difficulty measures are high. The *data structures* topic seems to be most distinct, with the lowest average solving percentage, highest time and highest actual time spend. Its solving percentage is significantly different from all other topics shown here under $\alpha=0.05$. This means this type of problems is solved less and later in the competition, taking more time. There must be kept in mind that in this particular set, it this might be that these particular more complicated problems could fall under the *data structure* topic by chance. In contrast, mathematical problems seem to be solved at an earlier timepoint, faster and with more efficiency; they are significantly different from *data structure* problems also at the average timepoint ($p=0.0004$). *Implementation* problems appear to be the most diverse, with the highest relative deviations. All the other topics have deviations too high to draw strong conclusions.

Topics and description

Another relevant piece of information for the contestants is to know how a topic relates to its description in order to more quickly identify the type of problem they are dealing with (e.g. are math problems always described with lots of text? Or are geometry problems illustrated with many images?). Table 16 below shows the most interesting metrics per topic, where differences between topics can be seen.

Topic	Words	std	Numbers	std	Images	std	Time limit	std
<i>Data structures</i>	460	234.48	67	98.84	0.700	0.675	5.625	2.925
<i>Graphs</i>	435	170.23	34	24.07	1.064	1.289	7.571	7.834
<i>Implementation</i>	364	139.99	31	28.82	0.625	0.792	6.130	11.491
<i>Math</i>	268	122.20	52	44.65	0.214	0.418	3.536	1.953
<i>Strings</i>	307	119.40	32	36.00	0.182	0.395	4.429	2.336
<i>Trees</i>	362	108.19	26	28.38	0.643	0.633	7.500	8.141
<i>Brute force</i>	344	127.17	35	32.43	0.543	0.721	5.524	6.025

Table 16: Descriptive metrics per topic.

Problems about *data structures* use on average the most words, which is significantly different from problems about *math*, *strings* and *brute force* under $\alpha=0.05$. So even though this topic has the most text describing it, it is on average the most difficult. Looking at the other statistics, *trees* use the least amount of numbers in its description and *data structures* the most, but these and other differences are not significant. The numbers in the text therefore are not distinctively informative. Furthermore, more images are used in *graphs* problems on average, which is to be expected as an illustration of a graph is more logical than, for instance, a *string* problem, which is more often described just in text. Problems about *graphs* use significantly more images than problems about *strings* and *math* under $\alpha=0.002$. Finally, the time limit indicates something about how long a solution can run and how efficient it must be, although this is entirely contextually dependent, as described in section 6.1, and therefore less informative on its own. There are differences between topics on the time limit, but this is more likely to be inherent to the sort of algorithms used and their speed.

Difficulty and description

Finally, the difficulty and attributes from the descriptions are compared. Looking at the correlation between these values independent from the topics, some weakly positive (but significant under $\alpha=0.001$) relations can be seen between the number of pages and the average time of solving (0.26) and between the number of words and the average time (0.35), indicating that more text is associated with a later time of solving. These are the two most remarkable correlations, although there are some other weak and less significant ($\alpha=0.05$) correlations present, for example between the time limit and the average time of solving (0.2).

Chapter 7: Conclusion

This chapter concludes the thesis and summarizes all findings by answering the main research question which is stated as follows:

MRQ: What winning strategies and characteristics can be identified in the ICPC data in order to enhance the programming competition community with knowledge about patterns existing throughout the years?

In order to answer this question, as it consists out of multiple components, an explicit answer must be given to each of the research questions as posed in section 2.4. The most important findings for each question are described below.

RQ1: What are the general patterns in the previous years?

An explorative look was taken at the patterns existing for the different competitions throughout the recent years. Each competition was summarized using two measures; popularity and difficulty, where for the latter a model to visualize a single competition year was made. First, general patterns found in these visualizations were explained, where problem characteristics such as spread-, solutions- and distribution of solutions over time where related to difficulty. Then, general patterns for each competition and noteworthy findings are described for each competition separately, which was done by stating statistics and semantically exploring the visualizations. Most interesting was looking at the World Finals, where a trend of multi-year balancing of the problem set became apparent, with some years clearly having more difficult problems than others. The European region's five competitions show a consistent number of countries participating, and growth or stability in the number of teams. No major fluctuations in difficulty seem to be present in the recent years, but some different types of trends and interesting numbers were found. The Latin American region is different from the others in the sense that it essentially is one big competition with multiple sites, which therefore has more data. Brazilian and Cuban teams seem to do the best on average in this region. Also, the solution graphs consistently show two similar patterns for all years, with a peak of easier problems being solved fast in the beginning and another problem being solved a lot over the course of each competition year. The South Pacific has the lowest number of around twelve teams participating in the most recent years, as the finals became invite only. A trend were teams are solving more problems can be seen in the graphs. Finally, the North American region has multiple competitions with major differences between them, in terms of in- or de-creasing popularity and difficulty, and also in trends and problem characteristics.

RQ2: What are characteristics of winning teams?

There was chosen to define winning teams as the teams that end up in the top 10 of a competition. These teams have distinctive traits that make them different from other teams. Starting with their demographic, these top teams originate from a limited number of countries and universities. There are only a few very good performing universities (and thus countries) with a relatively high number of teams ending up with a top 10 rank, as there is a steep curve for the representation of universities within all competitions. Performance wise, top10 teams use less attempts (also less attempts to solve) per problem, with a higher average solving ratio.

Furthermore, they are able to solve more problems in one attempt, making them slightly more efficient. Besides showing that the distribution of problems solved is different between the top group and other teams, the higher average is also caused by the fact that these top teams solve a higher proportion of more difficult problems. Another interesting finding is that top teams are very often the first to solve a problem, where the winning teams (with rank 1) alone were the first to solve a quarter of the problems in this dataset. On the subject of time, top 10 teams consistently solve most problem in less time. This can also be seen in the actual time spend per problem, which is again on average lower for the top teams. In conclusion, on all given comparative factors, the top10 teams outperform the other teams.

RQ3: Are there differences in regional, or regional and world final data?

Comparative metrics show that there are indeed differences between regions and also with the World Finals. It is important here is to keep in mind the presence of confounding variables such as problem difficulty and skill level of teams, of which the influence cannot be directly determined. The World Finals were used as a baseline, where was found that the European competitions in general show more similar numbers to the World Finals than the other competitions in terms of average solving percentage per problem (indicating average difficulty) and average percentage of problems solved per team (indicating the skill level of the average team in a competition). Furthermore, the World Finals were found to have the highest mean timepoint of solving, which indicates that this competition is the hardest of all. Then, countries were compared, where was shown that there are clear differences in the performances of teams within the World Finals, where Russian and Polish teams perform the best. Furthermore, the European competitions NEERC and NWERC were most similar to the World Finals. Using the visualizations made from each year for RQ1, it was found that the Latin American's solution distributions also look similar, although this competition appears to have more easier problems.

RQ4: What insights can be gained from the problem descriptions?

Two different ways of analyzing the problem descriptions were performed, focusing on two different aspects of the text. First, basic meta-information was extracted, where for example was found that the World Finals uses the most words in their descriptions. Then, by looking at the actual meaning of the text, expert labeling and machine learning were used to derive algorithmic topics. A platform was built to accommodate expert labeling, but submissions were limited too limited and only about a small part of the data, which could not be used to derive more insights. The machine learning techniques yielded more result. First the unsupervised Latent Dirichlet Allocation algorithm was run and optimized, after which an expert could label the identified topics. This led to the conclusion that some topics are more easily identifiable than others, because some are more hidden in a story that cannot be identified by simply looking at the words. Subsequently, a similar dataset from the CodeForces competitions was collected to turn the categorization problem into a supervised one and multiple different models were built to find the most generalizable one. The best model was then run on the ICPC data. Although this model is reasonably precise, its recall is low, which is why only a quarter of the problems could be labeled. Looking at the most prevalently tagged topics, it was found that problems about data structures were the most difficult in this dataset

and relatively solved later during the competition. There were also some other significant differences found between topics and the use of images. Other basic information was found not to be directly informative. More analysis would be possible if more descriptions were gathered or with better models, but using the limited generated data, only some aspects and topics could be looked at.

In conclusion, multiple patterns exist throughout the years. A retrospective description of all years resulted in an overview of events, similarities and differences within and between competitions. These were subsequently compared, looking at regions and top teams. Finally, problem descriptions were analyzed with machine learning algorithms, providing a first insight into the relation between the characteristics of problems, specifically looking at topics and other measures such as difficulty. All information found and derived could be used by multiple stakeholders involved in the competition.

Chapter 8: Discussion

This chapter aims to position the results by showing its limitations and exploring further future research that could be conducted. It also details findings from the research process which are relevant for other researchers doing similar work.

Limitations

The analysis performed and described in this thesis has some limitations. First, the created ICPC dataset was collected and initially cleaned using several special made scripts, but after this some extensive manual cleaning had to be performed due to bugs in the structuring process. Also, not all cleaning could be done automatically. The code created for this task was improved afterwards, but this makes the possibility to (re-)create the dataset less easy and the overall research less reproducible. Furthermore, not all regions were included when collecting scoreboards, which could have resulted in a limited view of the existing patterns. However, as conclusions are drawn while keeping in mind that only the data at hand, namely western regions, are considered, this limitation is of lesser impact.

As the analysis and description of the regions and years has an explorative nature, it is semantically inclined. This means that it is limited to the vision of the researcher, so it is possible that some findings or trends were missed. Also, confounding variables play a role in the interpretation of the findings, which made it sometimes harder to draw strong conclusions.

Finally, the best supervised model built to identify topics was relatively accurate but has a low average recall and thus is less generalizable. Furthermore, the amount of data gained from running this model is only limited. Conclusions drawn on the topics of the problems created using the model could therefore be less valid.

Research notes

Key notes gained from experience during this research project, which are relevant for similar data research, are mentioned here. First, during the ETL process of the dataset, there constantly was a choice between writing code to perform a certain task or doing it manually. Sometimes, the process was sped up by doing it semi-manual, but in hindsight, it was better to have done everything as automated and documented as possible, as this would have made the research more reproducible. Second, several times, text had to be rewritten because bugs were found in the data, making the statistics and conclusions drawn invalid and having to rewrite them. Very extensively checking the data for errors before doing analysis could have avoided this. Third, for the topic model building phase, many different algorithms and implementations thereof were tried. In hindsight, time could have been saved by doing less experiments, as these yielded in no significant improvement of the models and conclusions. The very specific context of finding hidden algorithmic topics in text should have better been considered before trusting in the power of machine learning to solve all problems. Fourth, as the analysis was largely centered around querying the data in different formats, saving all queries greatly improved the reproducibility of the results, which took almost no additional time. Fifth, the expected willingness from stakeholders in the domain of the ICPC and competitive programming to provide information was greatly overestimated. Significant efforts were

made to setup a platform that was only used for limited submissions. Setting the right expectations for the number of submissions and likeliness of getting enough input before spending much time on building a submission platform, would ensure if it is worth doing. Finally, data was analyzed using a mix of R and Python. It was found that different algorithms could be more quickly tried using Python. Using Python implementations sped up the research process.

Future work

This work has several different directions in which it can be expanded. These directions can be categorized into three different types of expansions; direct expansion, where the scope is broadened but the analysis stays the same, indirect expansion, where research that is related but fell outside the current scope is highlighted, and relevant ideas, which are interesting research opportunities found during the analysis that are worth to explore. Ideas for each of these ways are given below.

Direct expansion of this research is possible by including more information and doing the same analysis. An example of this is broadening the scope to include more ICPC regions such as Asia or Afrika, as it would be interesting to see how these regions relate to the other more western regions considered in this research. The different competitions of these regions could then also be analyzed in the same way as has been done here, to get an even more complete view of the recent history of the ICPC.

Another way to expand is by improving upon the topic learning chapter. This can be done by collecting and including more labeled problem sets besides only the CodeForces data (e.g. by including data from CodeChef¹⁷), which in turn could result in better and more generalizable models. As the number of tagged problems and identified topics are currently low, the focus should be on generating more data first. Running this (improved) model on more ICPC problem descriptions (beside only the European, Latin American, South Pacific and World Final problems), for instance by including the North American region, would result in a more complete analysis of the relation between the topics, difficulty level and team's skills within and between regions. Also, by relating the team's demographic background with topic information, educative insights could be gained, e.g. teams of university A could be better at dynamic programming problems but perform worse in tree algorithms, which is both helpful knowledge for the teams of that university as the university's educative program.

Furthermore, experts could be utilized even more than was done during this research. Doing structured interviews with multiple experts could result in deeper knowledge on the different topics analyzed during this thesis, such as strategies used during the competitions, and more real-life practical tips that could be helpful for the competitors. Also, more details about certain peculiarities and more history about the competitions could then be included. Furthermore, due to the limited number of submissions gained during the topic modelling, expert input could not be used to its fullest potential, e.g. difficulty ratings given by the experts are not used at all. Provided data could be expanded by asking more input from the experts,

¹⁷ <https://www.codechef.com/>

as the platform for this is already built. Also, expert submission data received up till now could be used as the basis for more analysis on the problems, which could for example be posed as a case study on the Latin American region (as most inputs are for this region only). Experts could then be consulted to create a comprehensive overview of this region, which could be interesting for the organization and competing teams.

Other future work lays in opportunities that arose during this research, but for which no time could be found to include. The main opportunity that was identified but excluded due to time constraints is collecting well established university ranking scores (e.g. QS World University Rankings¹⁸) and relating these scores to the performance of certain teams and thus universities. There could possibly be a relation where (perceived) better education could result in stronger and better performing teams, which could be interesting to know for educative purposes.

Another research direction not explored is predicting the final rank of teams at a certain point during the competition, based on their meta-information, their performance so far and competition meta-data. This information be used as a tool during competitions to give teams more insights in how well they are doing and if they should maybe change their strategy.

A final idea for future work is looking more in-depth at each of the submissions teams hand in at code-level. For example, the more recent World Finals have kept track of the actual different versions of code for each submission. Insights into common errors, different ways of coding and its effectiveness, and an overview of each attempt over time can be gained that way.

¹⁸ <https://www.topuniversities.com/subject-rankings/2019>

References

- About ICPC. (n.d.). Retrieved from <https://icpc.baylor.edu/regionals/abouticpc>
- About the Contest Problems. (n.d.). Retrieved from <https://icpc.baylor.edu/compete/problems>
- Abualigah, L. M., & Khader, A. T. (2017). Unsupervised text feature selection technique based on hybrid particle swarm optimization algorithm with genetic operators for the text clustering. *The Journal of Supercomputing*, 73(11), 4773-4795.
- Amraii, S. A. (2007). Observations on teamwork strategies in the ACM international collegiate programming contest. *Crossroads*, 14(1), 9.
- Arefin, A. S. (2006). Art of Programming Contest. *C Programming Tutorials, Data Structures and Algorithms*. ISBN, 984-32.
- Azevedo, A. I. R. L., & Santos, M. F. (2008). KDD, SEMMA and CRISP-DM: a parallel overview. *IADSDM*.
- Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(1), 993-1022.
- Bloomfield, A., & Sotomayor, B. (2016, February). A programming contest strategy guide. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*(pp. 609-614). ACM.
- Bora, A., & Sinha, A. (n.d.) Predicting algorithmic approach for programming problems from natural language problem description.
- Bowring, J. F. (2008, March). A new paradigm for programming competitions. In *ACM SIGCSE Bulletin* (Vol. 40, No. 1, pp. 87-91). ACM.
- Buckland, M., & Gey, F. (1994). The relationship between recall and precision. *Journal of the American society for information science*, 45(1), 12-19.
- Burton, B. A. (2007). Informatics olympiads: Approaching mathematics through code. *Mathematics Competitions*, 20(2), 29-51.
- Chatfield, C. (2000). *Time-series forecasting*. Chapman and Hall/CRC.
- Chollet, F. (2018). Keras: The python deep learning library. *Astrophysics Source Code Library*.
- Chuang, J., Manning, C. D., & Heer, J. (2012, May). Termite: Visualization techniques for assessing textual topic models. In *Proceedings of the international working conference on advanced visual interfaces* (pp. 74-77). ACM.
- Constantinescu, Z., Nicoara, S., Vladoiu, M., & Moise, G. (2017, September). Computer science student contests: individuals or teams. In *Proceedings of the 16th RoEduNet International Conference: Networking in Education and Research, Petru-Maior University of Targu Mures, Romania*.
- Dekel, O., & Shamir, O. (2010, March). Multiclass-multilabel classification with more classes than examples. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (pp. 137-144).
- Denny, M. J., & Spiriling, A. (2018). Text preprocessing for unsupervised learning: Why it matters, when it misleads, and what to do about it. *Political Analysis*, 26(2), 168-189.
- Ernst, F., Moelands, J., & Pieterse, S. (1996). Teamwork in programming contests: 3*1=4. *XRDS: Crossroads, The ACM Magazine for Students*, 3(2), 17-19.
- Fadel, A. (2018). CodeForces problems in PDF files, Github repository <https://github.com/AliOsm/PDF-CodeForces-Problems>
- Forišek, M. (2010, January). The difficulty of programming contests increases. In *International Conference on Informatics in Secondary Schools-Evolution and Perspectives* (pp. 72-85). Springer, Berlin, Heidelberg.

REFERENCES

- Godbole, S., & Sarawagi, S. (2004, May). Discriminative methods for multi-labeled classification. In *Pacific-Asia conference on knowledge discovery and data mining* (pp. 22-30). Springer, Berlin, Heidelberg.
- Grimmer, J., & Stewart, B. M. (2013). Text as data: The promise and pitfalls of automatic content analysis methods for political texts. *Political analysis*, 21(3), 267-297.
- Halim, S., & Halim, F. (2013). *Competitive Programming 3*. Lulu Independent Publish.
- Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.
- Katakis, I., Tsoumakas, G., & Vlahavas, I. (2008, September). Multilabel text classification for automated tag suggestion. In *Proceedings of the ECML/PKDD* (Vol. 18).
- Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.
- Maiatin, A., Mavrin, P., Parfenov, V., Pavlova, O., & Zubok, D. (2015). The Estimation of Winners' Number of the Olympiads' Final Stage. *Olympiads in Informatics*, 139.
- Manne, F. (2000). Competing in computing. *SIGCSE BULLETIN*, 32(3), 190-190.
- Manzoor, S. (2001). Common mistakes in online and real-time contests. *Crossroads*, 7(5), 4.
- Manzoor, S. (2006). Analyzing programming contest statistics. *Perspectives on Computer Science Competitions for (High School) Students*, 48.
- Manzoor, S. (2008). Common mistakes in online and real-time contests. *XRDS: Crossroads, The ACM Magazine for Students*, 14(4), 10-16.
- Niaksu, O. (2015). CRISP data mining methodology extension for medical domain. *Baltic Journal of Modern Computing*, 3(2), 92.
- Onwubolu, G. (2009). An inductive data mining system framework. In *Proceedings of the International Workshop on Inductive Modeling (IWIM'09)* (pp. 108-113).
- Paxton, J. (2007). Programming competition problems as a basis for an algorithms and data structures course. *Journal of Computing Sciences in Colleges*, 23(2), 27-32.
- Piatetsky-Shapiro, G. (2014, 10). CRISP-DM, still the top methodology for analytics, data mining, or data science projects. Retrieved from <http://www.kdnuggets.com/2014/10/crisp-dm-top-methodology-analytics-data-mining-data-science-projects.html>
- Poucher, B. (2017). *Bill Poucher's answer to What is more important to companies like Google, competitive programming skills or work experience in programming?* Retrieved from: <https://www.quora.com/What-is-more-important-to-companies-like-Google-competitive-programming-skills-or-work-experience-in-programming/answer/Bill-Poucher>
- Revilla, M. A., Manzoor, S., & Liu, R. (2008). Competitive learning in informatics: The UVa online judge experience. *Olympiads in Informatics*, 2, 131-148.
- Ribeiro, P., & Guerreiro, P. (2008). Early introduction of competitive programming. *Olympiads in Informatics*, 2, 149-162.
- Rosner, F., Hinneburg, A., Röder, M., Nettling, M., & Both, A. (2014). Evaluating topic coherence measures. *arXiv preprint arXiv:1403.6397*.
- Santos, A., Canuto, A., & Neto, A. F. (2011). A comparative analysis of classification methods to multi-label tasks in different application domains. *Int. J. Comput. Inform. Syst. Indust. Manag. Appl*, 3, 218-227.
- Schofield, A., Magnusson, M., & Mimno, D. (2017, April). Pulling out the stops: Rethinking stopword removal for topic models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers* (pp. 432-436).
- Shafique, U., & Qaiser, H. (2014). A comparative study of data mining process models (KDD, CRISP-DM and SEMMA). *International Journal of Innovation and Scientific Research*, 12(1), 217-222.

REFERENCES

- Shilov, N. V., & Yi, K. (2002). Engaging students with theory through acm collegiate programming contest. *Communications of the ACM*, 45(9), 98-101.
- Sievert, C., & Shirley, K. (2014). LDAvis: A method for visualizing and interpreting topics. In *Proceedings of the workshop on interactive language learning, visualization, and interfaces* (pp. 63-70).
- The ICPC Foundation. (2018). *ICPC Fact Sheet* [Fact sheet]. Retrieved from <https://icpc.baylor.edu/worldfinals/pdf/Factsheet.pdf>
- Trotman, A., & Handley, C. (2008). Programming contest strategy. *Computers & Education*, 50(3), 821-837.
- Tsoumakas, G., & Katakis, I. (2007). Multi-label classification: An overview. *International Journal of Data Warehousing and Mining (IJDWM)*, 3(3), 1-13.
- Verhoeff, T. (1997). The role of competitions in education. *Future world: Educating for the 21st century*, 1-10.
- Wieringa, R. J. (2014). *Design science methodology for information systems and software engineering*. Springer.
- Willett, P. (2006). The Porter stemming algorithm: then and now. *Program*, 40(3), 219-223.
- Wirth, R., & Hipp, J. (2000, April). CRISP-DM: Towards a standard process model for data mining. In *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining* (pp. 29-39). Citeseer.
- Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th international conference on evaluation and assessment in software engineering* (p. 38).
- World Finals Rules for 2019 (2018, October 4). Retrieved from <https://icpc.baylor.edu/worldfinals/rules>
- Zhang, M. L., & Zhou, Z. H. (2007). ML-KNN: A lazy learning approach to multi-label learning. *Pattern recognition*, 40(7), 2038-2048.
- Zhang, T., & Oles, F. J. (2001). Text categorization based on regularized linear classification methods. *Information retrieval*, 4(1), 5-31.

Appendix A: Data sources

In this appendix, the sources from all gathered data are detailed. This list is given to ensure reproducibility of this research. However, this list is not continuously updated, and links could therefore become outdated.

Table A.1: World Final data sources

Competition	Year	Source
<i>World Finals</i>	2018	https://web.archive.org/web/20180424212750/https://icpc.baylor.edu/scoreboard/
	2017	http://static.kattis.com/icpc/wf2017/
	2016	http://board.acmicpc.info/icpc2016/board.php
	2015	http://board.acmicpc.info/icpc2015/board.php
	2014	http://board.acmicpc.info/icpc2014/board.php
	2013	http://board.acmicpc.info/icpc2013/board.php
	2012	http://board.acmicpc.info/icpc2012/board.php

Table A.2: European regionals data sources

Competition	Year	Source
<i>Central Europe</i>	2018	https://contest.felk.cvut.cz/18cerc/rank.html
	2017	https://contest.felk.cvut.cz/17cerc/rank.html
	2016	http://cerc.hsin.hr/2016/
	2015	http://cerc.hsin.hr/2015/
	2014	https://cerc.tcs.uj.edu.pl/2014/ranking.html
	2013	https://cerc.tcs.uj.edu.pl/2013/ranking.html
	2012	https://cerc.tcs.uj.edu.pl/2012/ranking.html
<i>Northeastern Europe</i>	2018	https://neerc.ifmo.ru/archive/2018.html
	2017	https://neerc.ifmo.ru/archive/2017.html
	2016	https://neerc.ifmo.ru/archive/2016.html
	2015	https://neerc.ifmo.ru/archive/2015.html
	2014	https://neerc.ifmo.ru/archive/2014.html
	2013	https://neerc.ifmo.ru/archive/2013.html
	2012	https://neerc.ifmo.ru/archive/2012.html
<i>Northwestern Europe</i>	2018	http://2018.nwerc.eu/
	2017	http://2017.nwerc.eu/
	2016	http://2016.nwerc.eu/
	2015	http://2015.nwerc.eu/
	2014	http://2014.nwerc.eu/
	2013	https://2013.nwerc.eu/en/results/scoreboard/
	2012	https://2012.nwerc.eu/en/results/scoreboard/
<i>Southeastern Europe</i>	2018	http://acm.ro/
	2017	https://web.archive.org/web/20171025160647/http://acm.ro
	2016	http://acm.ro/2016/

APPENDIX A: DATA SOURCES

	2015	http://acm.ro/2015/
	2014	http://acm.ro/2014/
	2013	http://acm.ro/2013/
	2012	http://acm.ro/2012/
<i>Southwestern Europe</i>	2018	https://swerc.eu/2018/theme/scoreboard/public/
	2017	https://swerc.eu/2017/theme/results/official/public/
	2016	https://swerc.eu/2017/theme/cached/2016/
	2015	https://swerc.eu/2017/theme/cached/2015/
	2014	https://swerc.eu/2017/theme/cached/2014/
	2013	https://swerc.eu/2017/theme/cached/2013/
	2012	https://swerc.eu/2017/theme/cached/2012/

Table A.3: Latin American regional data sources (*Note that all sub-competitions are stored in one scoreboard at the same website*)

Competition	Year	Source
<i>Latin America</i>	2018	http://maratona.ime.usp.br/resultados18/
	2017	http://www.bombonera.org/oldboards/score2017/score/
	2016	http://bombonera.org/oldboards/score2016/
	2015	http://bombonera.org/oldboards/score2015/
	2014	http://bombonera.org/oldboards/score2014/
	2013	http://bombonera.org/oldboards/score2013/
	2012	http://bombonera.org/oldboards/score2012/score2012/

Table A.4: South Pacific regional data sources

Competition	Year	Source
<i>South Pacific</i>	2018	http://public.webdev.aut.ac.nz/ACM/Scoreboards/2018/Regional/FinalScoreboard2018.html
	2017	http://public.webdev.aut.ac.nz/ACM/Scoreboards/2017/Regional/FinalScoreboard.htm
	2016	http://public.webdev.aut.ac.nz/ACM/Scoreboards/2016/Regional/FinalScoreboard.htm
	2015	http://public.webdev.aut.ac.nz/ACM/Scoreboards/2015/Regional/Scoreboard.htm
	2014	http://public.webdev.aut.ac.nz/ACM/Scoreboards/2014/Regional/Scoreboard.html
	2013	http://public.webdev.aut.ac.nz/ACM/Scoreboards/2013/Scoreboard.html
	2012	http://public.webdev.aut.ac.nz/ACM/Scoreboards/2012/Scoreboard.html

Table A.5: North American regional data sources (*Note that not all competitions are listed here*)

Competition	Year	Source
<i>East-Central NA</i>	2018	https://ecna18.kattis.com/standings/standalone
	2017	https://ecna17.kattis.com/standings/standalone
	2016	https://ecna16.kattis.com/standings/standalone
	2015	https://ecna15.kattis.com/standings/standalone
	2014	https://web.archive.org/web/20160828011045/http://acm-ecna.yzu.edu:80/PastResults/2014/standings.html
	2013	https://web.archive.org/web/20131115071843/http://icpc01.cc.yzu

APPENDIX A: DATA SOURCES

		edu:80/scoreboard/ 2012 https://web.archive.org/web/20150822192024/http://acm.ashland.edu:80/2012/standings.html
<i>Greater NY</i>	2018 2017 2016 2015 2014 2013 2012	http://acmgnyr.org/year2018/standings.shtml http://acmgnyr.org/year2017/standings.shtml http://acmgnyr.org/year2016/standings.shtml http://acmgnyr.org/year2015/standings.shtml http://acmgnyr.org/year2014/standings.shtml http://acmgnyr.org/year2013/standings.shtml http://acmgnyr.org/year2012/standings.shtml
<i>Mid-Atlantic USA</i>	2018 2017 2016 2015 2014 2013 2012	https://mausa18.kattis.com/standings https://mausa17.kattis.com/standings https://web.archive.org/web/20161109015551/http://midatl.radford.edu:80/scoreboard/summary.html https://web.archive.org/web/20160118135544/http://midatl.radford.edu:80/scoreboard/summary.html https://web.archive.org/web/20150519074453/https://www.cs.odu.edu/~zeil/icpc/scoreboard2014.html https://web.archive.org/web/20140401093853/http://www.radford.edu:80/~acm/midatl/2013_scoreboard.html http://midatl.radford.edu/docs/scoreboard/summary.html
<i>Mid-Central USA</i>	2018 2017 2016 2015 2014 2013 2012	https://mcpc18.kattis.com/standings https://mcpc17.kattis.com/standings https://mcpc16.kattis.com/standings https://mcpc15.kattis.com/standings N/A N/A N/A
<i>NA Invitational</i>	2018 2017 2016 2015 2014 2013 2012	https://naipc18.kattis.com/standings/standalone https://naipc17.kattis.com/standings/standalone https://naipc16.kattis.com/standings/standalone https://naipc15.kattis.com/standings/standalone http://naipc.uchicago.edu/2014/scoreboard-final-onsite.html http://icpc.cs.uchicago.edu/invitational2013/board_final.html http://icpc.cs.uchicago.edu/invitational2012/scoreboard.html
<i>North-Central NA</i>	2018 2017 2016 2015 2014 2013 2012	https://ncna18.kattis.com/standings https://ncna17.kattis.com/standings http://cse.unl.edu/~upe/contest/ http://cse.unl.edu/~upe/contest/ http://cse.unl.edu/~upe/contest/ http://cse.unl.edu/~upe/contest/ http://cse.unl.edu/~upe/contest/
<i>Pacific North-West</i>	2018	http://acmicpc-pacnw.org/scoreboard/2018/index1.html

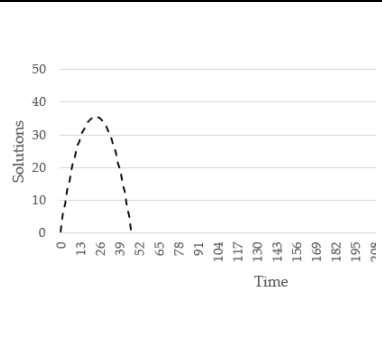
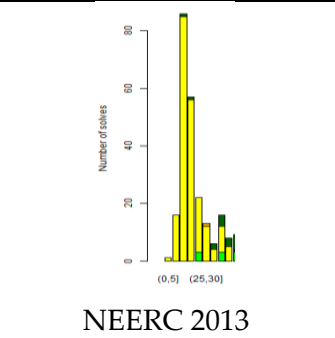
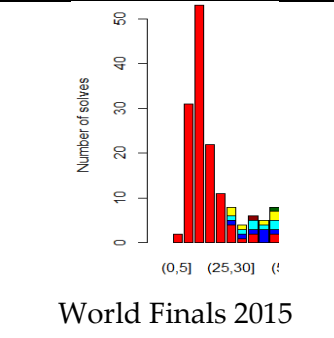
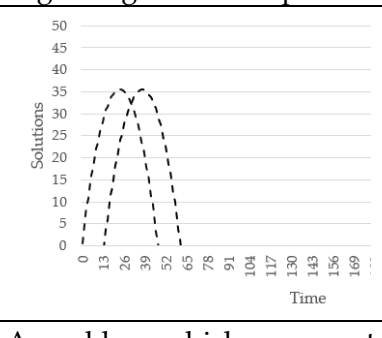
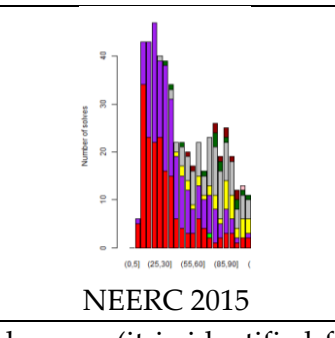
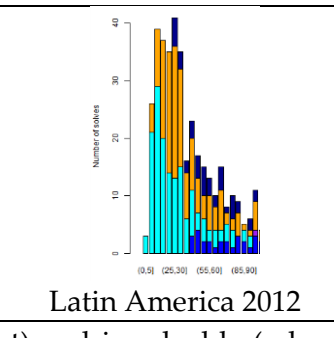
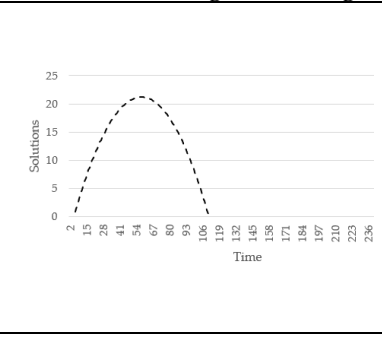
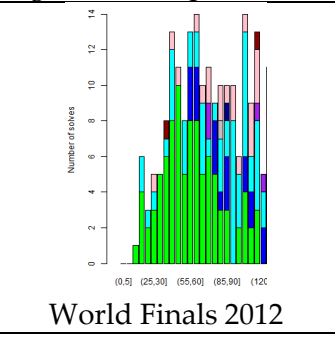
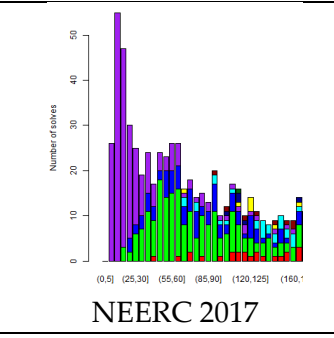
APPENDIX A: DATA SOURCES

	2017	http://acmicpc-pacnw.org/ProblemSet/2017/index1.html
	2016	https://web.archive.org/web/20170111143951/http://www.acmicpc-pacnw.org:80/scoreboard/index1.html
	2015	http://acmicpc-pacnw.org/ProblemSet/2015/index1.html
	2014	http://acmicpc-pacnw.org/ProblemSet/2014/html.all/index1.html
	2013	http://acmicpc-pacnw.org/ProblemSet/2013/index.html
	2012	http://acmicpc-pacnw.org/Standings/2012/index.html
<i>Rocky mountain</i>	2018	https://rmc18.kattis.com/standings
	2017	https://rmc17.kattis.com/standings
	2016	https://rmc16.kattis.com/standings
	2015	https://rmc15.kattis.com/standings
	2014	https://org.coloradomesa.edu/~wmacevoy/rmrc/2014/scoreboard.html
	2013	https://org.coloradomesa.edu/~wmacevoy/rmrc/2013/scoreboard_byrank.html
	2012	https://web.archive.org/web/20130913013048/http://org.coloradomesa.edu:80/acm/rmrc/2012/scoreboard_byrank.html
<i>South-Central USA</i>	2018	http://ld2018.scusa.lsu.edu/standings-contest/
	2017	http://ld2017.scusa.lsu.edu/scoreboard-regional/
	2016	http://ld2016.scusa.lsu.edu/scoreboard-regional/
	2015	http://ld2015.scusa.lsu.edu/scoreboard-regional/
	2014	http://ld2014.scusa.lsu.edu/scoreboard-regional/
	2013	http://ld2013.scusa.lsu.edu/scoreboard-regional/
	2012	http://ld2012.scusa.lsu.edu/scoreboard-regional/
<i>South-East USA</i>	2018	https://ser.cs.fit.edu/ser2018/ser2018-results-div1.pdf
	2017	https://ser.cs.fit.edu/ser2017/ser2017-results-div1.pdf
	2016	https://ser.cs.fit.edu/ser2016/ser2016-results-div1.pdf
	2015	https://ser.cs.fit.edu/ser2015/ser2015-results-div1.pdf
	2014	N/A
	2013	https://ser.cs.fit.edu/ser2013/ser2013_final_standingsI.pdf
	2012	https://ser.cs.fit.edu/ser2012/ser2012_scoreboard.pdf
<i>Southern California</i>	2018	http://socalcontest.org/history/2018/Scoreboard-2018.shtml
	2017	http://socalcontest.org/history/2017/details-2017.shtml
	2016	http://socalcontest.org/history/2016/details-2016.shtml
	2015	http://socalcontest.org/history/2015/details-2015.shtml
	2014	http://socalcontest.org/history/2014/details-2014.shtml
	2013	http://socalcontest.org/history/2013/details-2013.shtml
	2012	http://socalcontest.org/history/2012/details-2012.shtml

Appendix A.2: Sources for problem pdfs

Competition	Year	Source
<i>Europe, World Finals, South Pacific & Latin America</i>	2012- 2017	https://icpcarchive.ecs.baylor.edu/ index.php?option=com_onlinejudge&Itemid=8
<i>CEERC</i>	2018	https://contest.felk.cvut.cz/18cerc/solved.html
<i>NEERC</i>	2018	https://neerc.ifmo.ru/archive/2018/neerc-2018-statement.pdf
<i>NWERC</i>	2018	http://2018.nwerc.eu/files/nwerc2018problems.pdf
<i>SEERC</i>	2018	http://acm.ro/2018/index.html
<i>SWERC</i>	2018	https://swerc.eu/2018/theme/problems/swerc.pdf
<i>World Finals</i>	2018	https://icpc.baylor.edu/worldfinals/problems/icpc2018.pdf
<i>Latin America</i>	2018	http://maratona.ime.usp.br/resultados18/contest_onesided.pdf
<i>South Pacific</i>	2018	http://public.webdev.aut.ac.nz/ACM/ACM_ProblemSets/2018/yyRegionals.pdf

Appendix B: Patterns in problems

Pattern	Example distribution	Example 1	Example 2	
1	A large spike of the same problem being solved in the beginning, and (almost) all solves are clustered around this time. This is a problem which both appears to be easy as it is identified by most teams, and actually is relatively easy as the timespan for solving is short and the number of solves is high. The frequency at which this pattern occurs differs per competition.	 <p>A line graph with 'Solutions' on the y-axis (0 to 50) and 'Time' on the x-axis (0 to 195). A dashed line shows a single peak starting at time 0, reaching a maximum of approximately 35 solutions around time 26, and returning to 0 by time 52.</p>	 <p>A stacked bar chart with 'Number of solves' on the y-axis (0 to 80) and time intervals on the x-axis: (0,5], (25,30]. The first bar (0,5] is very tall, reaching over 80 solves, while the second bar (25,30] is significantly shorter, around 20 solves.</p>	 <p>A stacked bar chart with 'Number of solves' on the y-axis (0 to 50) and time intervals on the x-axis: (0,5], (25,30]. The first bar (0,5] is very tall, reaching over 50 solves, while the second bar (25,30] is much shorter, around 10 solves.</p>
2	A large spike of the multiple problems being solved in the beginning, where most solves are clustered around this time. This pattern is very similar to pattern 1, but shows overlap of more than one (usually two) problems both being solved in the beginning of the competition.	 <p>A line graph with 'Solutions' on the y-axis (0 to 50) and 'Time' on the x-axis (0 to 169). Two dashed lines show overlapping peaks. The first peak starts at time 0, reaches a maximum of about 35 around time 26, and ends at time 52. The second peak starts at time 13, reaches a maximum of about 35 around time 39, and ends at time 65.</p>	 <p>A stacked bar chart with 'Number of solves' on the y-axis (0 to 45) and time intervals on the x-axis: (0,5], (25,30], (55,60], (85,90]. Multiple overlapping bars of different colors show a high concentration of solves in the first interval, with a secondary smaller peak in the second interval.</p>	 <p>A stacked bar chart with 'Number of solves' on the y-axis (0 to 40) and time intervals on the x-axis: (0,5], (25,30], (55,60], (85,90]. Multiple overlapping bars of different colors show a high concentration of solves in the first interval, with a secondary smaller peak in the second interval.</p>
3	A problem which appears to be easy (it is identified fast) and is solvable (a large group of solves and also a large proportion of teams that solved it), but takes more time depending on the team. This is a pattern where you see a cluster of solutions with a peak timepoint in the middle, but more spread out. The spread considered here is occurring at the beginning of the competition.	 <p>A line graph with 'Solutions' on the y-axis (0 to 25) and 'Time' on the x-axis (2 to 236). A dashed line shows a broad peak starting at time 2, reaching a maximum of about 22 around time 67, and returning to 0 by time 106.</p>	 <p>A stacked bar chart with 'Number of solves' on the y-axis (0 to 14) and time intervals on the x-axis: (0,5], (25,30], (55,60], (85,90], (120,125]. The bars are spread out across the middle of the competition, with a peak around the (55,60] interval.</p>	 <p>A stacked bar chart with 'Number of solves' on the y-axis (0 to 50) and time intervals on the x-axis: (0,5], (25,30], (55,60], (85,90], (120,125], (160,165]. The bars are spread out across the middle of the competition, with a peak around the (55,60] interval.</p>
4	For almost all years and competitions, there are problems that are not solved often, but for which solutions appear sporadically during the whole competition. For this pattern, there is no apparent peak or time at which the problem is solved. The time at which the first solution for such a problem is found is not directly at the beginning.			

APPENDIX B: PATTERNS IN PROBLEMS

		<p>Latin America 2013, cyan</p>	<p>East-Central NA 2017, yellow</p>
<p>5</p>	<p>This same sporadic pattern also occurs for problems which are solved in relatively higher amounts at no specific timepoint. Again, the solutions for this problem do not directly appear at the start, but later. This pattern is characterised by the continuous appearance of solutions starting often after an initial peak of solutions. The occurrence of this pattern is less often.</p>		
		<p>East-Central NA 2013, blue</p>	<p>Latin America 2016, cyan</p>
<p>6</p>	<p>As sporadic patterns are seen the most in the data, another one is listed here. The difference with the previous ones however is that the problem is identified and solved from (almost or at) the start of the competition till the end, over the whole duration of the competition. Many more similar examples exist, but these problems are all characterized by the fact that they are not grouped and scattered over time, with no specific peaks.</p>		
		<p>Latin America 2013, blue</p>	<p>East-Central 2016, green</p>
<p>7</p>	<p>Another relatively common pattern is one where problems are solved by a small proportion of teams and solutions only appear at the end. These are most likely difficult problems saved for last.</p>		
		<p>World 2017, green</p>	<p>North-Central NA 2013, red</p>

Appendix C: Solution distribution graphs

Have been relocated to the supplementary github page <https://github.com/RickdeBoer/ICPC-Thesis> due to space concerns.

Appendix D: Model scores

	precision	recall	f1-score	support
binary search	0.18	0.40	0.25	30
bitmasks	0.00	0.00	0.00	14
brute force	0.22	0.41	0.28	61
combinatorics	0.53	0.43	0.48	23
constructive algorithms	0.21	0.35	0.27	43
data structures	0.36	0.61	0.45	67
dfs and similar	0.36	0.55	0.43	33
divide and conquer	0.25	0.12	0.17	8
dp	0.36	0.65	0.46	85
dsu	0.29	0.11	0.16	18
geometry	0.52	0.62	0.57	21
graphs	0.46	0.56	0.51	32
greedy	0.34	0.63	0.44	81
hashing	0.50	0.17	0.25	6
implementation	0.38	0.72	0.50	120
math	0.31	0.64	0.42	73
number theory	0.48	0.50	0.49	26
probabilities	0.75	0.55	0.63	11
shortest paths	0.25	0.10	0.14	10
sortings	0.29	0.34	0.31	47
strings	0.44	0.65	0.53	23
trees	0.66	0.66	0.66	29
two pointers	0.17	0.12	0.14	16
micro avg	0.34	0.53	0.42	877
macro avg	0.36	0.43	0.37	877
weighted avg	0.35	0.53	0.41	877
samples avg	0.35	0.56	0.40	877

Figure D1: Logistic regression model evaluation metrics.

	precision	recall	f1-score	support
binary search	0.00	0.00	0.00	30
bitmasks	0.00	0.00	0.00	14
brute force	0.00	0.00	0.00	61
combinatorics	1.00	0.17	0.30	23
constructive algorithms	0.40	0.05	0.08	43
data structures	0.65	0.19	0.30	67
dfs and similar	0.47	0.27	0.35	33
divide and conquer	0.00	0.00	0.00	8
dp	0.60	0.07	0.13	85
dsu	0.25	0.06	0.09	18
geometry	0.63	0.57	0.60	21
graphs	0.60	0.47	0.53	32
greedy	0.50	0.11	0.18	81
hashing	0.50	0.17	0.25	6
implementation	0.58	0.43	0.50	120
math	0.51	0.27	0.36	73
number theory	0.46	0.23	0.31	26
probabilities	1.00	0.27	0.43	11
shortest paths	0.25	0.10	0.14	10
sortings	0.00	0.00	0.00	47
strings	0.56	0.65	0.60	23
trees	0.61	0.38	0.47	29
two pointers	0.00	0.00	0.00	16
micro avg	0.55	0.21	0.30	877
macro avg	0.42	0.19	0.24	877
weighted avg	0.45	0.21	0.26	877
samples avg	0.32	0.23	0.25	877

Figure D2: Multinomial Bayes model evaluation metrics.

	precision	recall	f1-score	support
binary search	0.09	0.03	0.05	30
bitmasks	0.00	0.00	0.00	14
brute force	0.41	0.11	0.18	61
combinatorics	0.71	0.22	0.33	23
constructive algorithms	0.37	0.16	0.23	43
data structures	0.58	0.31	0.41	67
dfs and similar	0.47	0.21	0.29	33
divide and conquer	0.00	0.00	0.00	8
dp	0.44	0.22	0.30	85
dsu	0.00	0.00	0.00	18
geometry	0.69	0.52	0.59	21
graphs	0.62	0.31	0.42	32
greedy	0.32	0.25	0.28	81
hashing	0.00	0.00	0.00	6
implementation	0.53	0.38	0.45	120
math	0.41	0.29	0.34	73
number theory	0.80	0.31	0.44	26
probabilities	0.86	0.55	0.67	11
shortest paths	1.00	0.10	0.18	10
sortings	0.36	0.09	0.14	47
strings	0.52	0.52	0.52	23
trees	0.65	0.38	0.48	29
two pointers	0.67	0.12	0.21	16
micro avg	0.48	0.25	0.33	877
macro avg	0.46	0.22	0.28	877
weighted avg	0.46	0.25	0.31	877
samples avg	0.36	0.27	0.29	877

Figure D3: Support Vector Machine model (gamma=0.001) evaluation metrics.

	precision	recall	f1-score	support
binary search	0.00	0.00	0.00	30
bitmasks	0.00	0.00	0.00	14
brute force	0.00	0.00	0.00	61
combinatorics	0.00	0.00	0.00	23
constructive algorithms	0.00	0.00	0.00	43
data structures	0.64	0.10	0.18	67
dfs and similar	0.57	0.12	0.20	33
divide and conquer	0.00	0.00	0.00	8
dp	0.67	0.07	0.13	85
dsu	0.00	0.00	0.00	18
geometry	1.00	0.38	0.55	21
graphs	0.73	0.25	0.37	32
greedy	0.57	0.05	0.09	81
hashing	0.00	0.00	0.00	6
implementation	0.68	0.30	0.42	120
math	0.61	0.15	0.24	73
number theory	0.80	0.15	0.26	26
probabilities	1.00	0.09	0.17	11
shortest paths	0.00	0.00	0.00	10
sortings	0.00	0.00	0.00	47
strings	0.46	0.26	0.33	23
trees	0.86	0.21	0.33	29
two pointers	0.00	0.00	0.00	16
micro avg	0.65	0.12	0.20	877
macro avg	0.37	0.09	0.14	877
weighted avg	0.46	0.12	0.18	877
samples avg	0.23	0.14	0.16	877

Figure D4: Support Vector Machine model (gamma=2) evaluation metrics.

APPENDIX D: MODEL SCORES

	precision	recall	f1-score	support		precision	recall	f1-score	support
binary search	1.00	0.03	0.05	38	binary search	0.15	0.27	0.20	30
dp	0.00	0.00	0.00	81	bitmasks	0.00	0.00	0.00	14
dsu	0.00	0.00	0.00	13	brute force	0.24	0.62	0.35	61
geometry	0.60	0.33	0.43	18	combinatorics	0.46	0.57	0.51	23
graphs	0.82	0.21	0.34	42	constructive algorithms	0.24	0.70	0.35	43
greedy	0.29	0.11	0.16	70	data structures	0.43	0.61	0.51	67
hashing	0.00	0.00	0.00	9	dfs and similar	0.42	0.55	0.47	33
implementation	0.50	0.41	0.45	128	divide and conquer	0.40	0.25	0.31	8
math	0.62	0.14	0.22	96	dp	0.34	0.74	0.46	85
number theory	0.00	0.00	0.00	30	dsu	0.45	0.28	0.34	18
probabilities	0.60	0.43	0.50	7	geometry	0.66	0.90	0.76	21
shortest paths	0.00	0.00	0.00	13	graphs	0.48	0.66	0.55	32
sortings	0.50	0.03	0.06	33	greedy	0.32	0.80	0.46	81
strings	0.50	0.32	0.39	25	hashing	0.20	0.17	0.18	6
trees	0.45	0.22	0.29	23	implementation	0.37	0.88	0.52	120
two pointers	0.00	0.00	0.00	9	math	0.30	0.81	0.43	73
bitmasks	0.00	0.00	0.00	14	number theory	0.36	0.54	0.43	26
brute force	0.00	0.00	0.00	68	probabilities	0.44	0.36	0.40	11
combinatorics	0.33	0.11	0.16	28	shortest paths	0.14	0.10	0.12	10
constructive algorithms	0.50	0.05	0.09	41	sortings	0.21	0.30	0.25	47
data structures	0.57	0.06	0.12	62	strings	0.45	0.78	0.57	23
dfs and similar	0.50	0.14	0.21	37	trees	0.66	0.66	0.66	29
divide and conquer	0.00	0.00	0.00	7	two pointers	0.00	0.00	0.00	16
micro avg	0.50	0.13	0.21	892	micro avg	0.33	0.64	0.44	877
macro avg	0.34	0.11	0.15	892	macro avg	0.34	0.50	0.38	877
weighted avg	0.40	0.13	0.18	892	weighted avg	0.34	0.64	0.43	877
samples avg	0.27	0.16	0.19	892	samples avg	0.34	0.68	0.43	877

Figure D4: Multilabel K-NN model evaluation metrics.

Figure D5: Neural network model evaluation metrics.

	precision	recall	f1-score	support
binary search	0.30	0.10	0.15	30
bitmasks	0.00	0.00	0.00	14
brute force	0.35	0.15	0.21	61
combinatorics	0.83	0.22	0.34	23
constructive algorithms	0.75	0.14	0.24	43
data structures	0.66	0.28	0.40	67
dfs and similar	1.00	0.09	0.17	33
divide and conquer	0.00	0.00	0.00	8
dp	0.76	0.15	0.25	85
dsu	1.00	0.11	0.20	18
geometry	1.00	0.24	0.38	21
graphs	0.73	0.34	0.47	32
greedy	0.43	0.16	0.23	81
hashing	1.00	0.33	0.50	6
implementation	0.55	0.39	0.46	120
math	0.53	0.27	0.36	73
number theory	0.73	0.31	0.43	26
probabilities	1.00	0.27	0.43	11
shortest paths	1.00	0.10	0.18	10
sortings	1.00	0.06	0.12	47
strings	0.52	0.65	0.58	23
trees	0.90	0.31	0.46	29
two pointers	0.00	0.00	0.00	16
micro avg	0.59	0.22	0.33	877
macro avg	0.65	0.20	0.29	877
weighted avg	0.63	0.22	0.31	877
samples avg	0.35	0.25	0.27	877

Figure D6: Final tuned model evaluation metrics.

APPENDIX D: MODEL SCORES

Topic	Best algorithm	Preprocessing settings
<i>Binary search</i>	Logistic Regression	RW:0, MF:5000, NG: 5
<i>Bitmasks</i>	N/A	N/A
<i>Brute force</i>	SVM (gamma:0.001)	RW:4, MF:5000, NG:4
<i>Combinatorics</i>	SVM (gamma:0.001)	RW:3, MF:8000, NG:5
<i>Constructive algorithms</i>	MNB	RW:2, MF:10000, NG:4
<i>Data structures</i>	Logistic Regression	RW:0, MF:8000, NG:3
<i>Dfs and similar</i>	SVM (gamma:2)	RW:0 MF:10000, NG:4
<i>Divide and conquer</i>	N/A	N/A
<i>DP</i>	MNB	RW:none, MF:5000, NG:5
<i>DSU</i>	NNET	WL:none, MF:5000, NG:0
<i>Geometry</i>	NNET	RW:4, MF:10000, NG:0
<i>Graphs</i>	MNB	RW:5, MF:5000, NG:5
<i>Greedy</i>	MNB	RW:4, MF:10000, NG:5
<i>Hashing</i>	MNB	RW:2, MF:10000, NG:4
<i>Implementation</i>	Logistic Regression	RW:1, MF:10000
<i>Math</i>	MNB	RW:3, MF:10000, NG:3
<i>Number theory</i>	Logistic Regression	RW:5, MF:5000
<i>Probabilities</i>	Logistic Regression	RW:3, MF:10000
<i>Shortest paths</i>	Logistic Regression	RW:none, MF:5000
<i>Sortings</i>	MNB	RW:1, MF:8000, NG:4
<i>Strings</i>	MNB	RW:1, MF:10000, NG:5
<i>Trees</i>	Logistic Regression	RW:0, MF:8000
<i>Two pointers</i>	SVM (gamma:0.001)	RW:none, MF:5000, NG:4

Table D1: Final model setup

Here, RW means Remove Words up till the given length X, with 0 meaning only removing dots, double spaces and control characters. MF means the maximum number of features used, and NG means the number N-grams considered.

Acknowledgements

A big thank you to the people who generously spend some of their time to provided information on the problems. Although not enough data could be collected, their information is used as a steppingstone and for comparison, and also is available in a file accompanying this thesis at the github page. Names below are provided by the experts themselves:

Javier Ojeda, Leonardo Tironi Fassini, Yuri Cardoso Santamarina, Bruno Almêda de Oliveira, "laderlappen", "Kallaseldor", Eduardo Cardozo Cantarym Pedro, "Thilio", "RenatoBA", Jong Young Lee, "juckter", Caique Coelho, "doughnobrega", André Winston, "Nson", "begv", Gerson Yesid Lázaro, "juniorandrade", "pauloamed", Lucas Lopes Pereira, Anderson, "edimais", Bixão da Goiaba and "Weiss".