

The ICPC Resolver



An ACM ICPC Tool

The *ICPC Resolver* is a tool for graphical animation of contest results. It shows the final runs submitted during a contest in an interesting way, and leads up to display of the award winners. The Resolver concept was created by Fredrik Niemela and Mattias de Zalenski at KTH Royal Technical University. The ICPC Tools Resolver implementation was developed by Tim deBoer of IBM Corporation.

The Resolver is designed to be used in contests where the scoreboard is "frozen" prior to the end of the contest - that is, where the result of runs submitted in the last part of the contest are not displayed on the scoreboard (such runs are typically marked as "pending"). The Resolver produces a dynamic display by stepping through ("resolving") pending runs and generating displays showing the contest winners in ranked order, along with citations for awards earned.

After displaying an introductory "splash screen", a single keystroke or mouse click causes the Resolver to display the contest standings as of the time the scoreboard was frozen. A key or mouse click then causes it to advance to the bottom of the standings; a subsequent key/click starts the "resolving" process: starting at the bottom, it moves up until it reaches a team that has one or more pending submissions during the freeze time. Each pending run is 'resolved' (to either a "yes" or "no" judgment), and if the run was successful the team will move 'up' into their new position based on the results.

Options allow you to configure when the resolver pauses, but by default it will continue moving up and resolving until it gets to an 'interesting' case - typically a first-to-solve award, a "group" or "region" winner, or a gold/silver/bronze award winner. When it reaches an award, the resolver will pause and switch to a screen showing the team name, the logo and image (if available from a CDS; see below), and an award citation. Once the award has been handed out, clicking returns to the regular Resolver screen and continues the resolving process.

A variety of options are available, including managing the speed at which the Resolver runs, controlling various "single-step" operations, configuring categories of awards to be acknowledged during the resolving process, and controlling simultaneous Resolver operations at multiple contest sites.

The following shows a screen-shot of the Resolver in action. Pending (unresolved) runs are shown in yellow; the team whose pending run is about to be "resolved" is highlighted. If it is a "Yes" then the team's entire row will move dynamically up the screen to their new position.

Rank	Name	Solved	Time
1	St. Petersburg National Research University of IT, Mechanics and Optics	10	1001
2	Moscow State University	10	1030
3	Tsinghua University	10	1234
4	University of California at Berkeley	10	1347
5	University of Zagreb	10	1501
6	Charles University in Prague	10	1567
7	Shanghai Jiao Tong University	10	1616
8	Massachusetts Institute of Technology	10	1629
9	The University of Tokyo	9	773
10	Peking University	9	939
11	Korea University	9	1220
12	University of Warsaw	9	1233

Operational Modes

The Resolver runs in one of two basic modes: *standalone* or *distributed*. In standalone mode, the Resolver is run on a single machine and displays its output on that machine's display screen. Typically a projector is connected to the graphics output port of the machine, allowing the audience to see the Resolver display as it reveals the results of the contest. The person running the standalone Resolver uses keyboard/mouse input to control the flow of the Resolver (for example, to advance to the "next run to be resolved" or to display "award results"). Standalone mode is typically used for example to display the results at the end of a contest held at a single site.

In distributed mode, the Resolver is run on several machines. One machine (the *presenter* client) acts as the controller for the flow of events (the Contest Administrator uses keyboard/mouse input to control Resolver operation, similar to stand-alone mode). A second machine runs a server; and one or more additional machines act as *viewer* clients. The server is started first and listens for connections from clients; presenter and viewer clients connect to the server through a configurable port.

Distributed mode is typically used in contests where teams are competing at different (physically remote) sites, or where the Contest Administrator wishes to see a different view of the resolving process than what is displayed to local and/or remote audiences. In distributed mode the Contest Administrator runs a presenter client and controls the flow of the Resolver, while one or more viewer clients show the current Resolver status screen. The presenter client has the option of displaying additional useful information to the person controlling the Resolver (for example, when a pending run is selected it can show a small dialog saying something like "if the team solves this run then they will jump up into 3rd place"). This extra information display is only visible on the presenter client; the viewer clients see only the main Resolver output.

Input Data Sources

The Resolver works by reading the *event feed* output of a *Contest Control System (CCS)* [The ICPC Resolver will work with any CCS that produces an event feed which is compliant with the *CLI CCS Specification* defined at <https://clics.ecs.baylor.edu/>. Systems known to produce compliant event feeds include [PC-Squared](#) and [Kattis](#); other Contest Control Systems may also produce compatible event feeds and hence work with the Resolver.]. The Resolver is capable of operating with event feed data obtained

from one of three different sources: an *event feed file*, a *contest data package* (folder), or a *Contest Data Server* [A *Contest Data Server* is another ACM ICPC Tool; see the [ACM ICPC website](#) for details].

An event feed file is created by connecting to port 4713 on a machine running a compatible CCS and saving the resulting output to a file (see https://clics.ecs.baylor.edu/index.php/Event_Feed for details on the structure of an event feed). Once the event feed data has been saved in a file, the Resolver can be started with the event feed file as an argument.

A second way to provide the Resolver with input data is by creating a *contest data package* (CDP). A CDP is an arbitrarily-named folder with specific contest-configuration contents. If the Resolver is started with its first argument being the path (relative or absolute) to a folder whose contents are organized following the [CLI Contest Data Package draft specification](#) it will read its event feed data from that folder [Note that the Resolver expects to find the event feed information stored in a file named *contest.xml* located in the CDP root folder when it reads from a CDP.] .

Alternatively, the Resolver can obtain its event feed data from a Contest Data Server (CDS). A CDS provides URL access points for obtaining event feed and other contest data; starting the Resolver with the URL of a CDS causes it to connect to the CDS and pull down the necessary information dynamically. (Note: the CDS must have been configured to connect to a compatible CCS in order for it to supply event feed data to the Resolver.)

The main reason for using either a CDP or a CDS instead of a simple event feed file to provide Resolver input is that a CDP or CDS can also provide other information to the Resolver. For example, if the CDP or CDS has been configured with team pictures and/or team school/university logos, the Resolver knows how to pull these from the corresponding input location as well and will use them in the output display. The sample image (above) shows the use of such university logos; these came from starting the Resolver by pointing it to a CDP (but could have just as easily been generated by starting the Resolver and having it connect to a CDS).

Note: The Resolver requires the input event feed to be "*finalized*". "Finalizing" is an operation performed in the CCS which generates the event feed, indicating that the contest has ended and the event feed contains the "complete (final) results".

Using the Resolver

Installation

To install the Resolver, download and unzip the distribution package to any convenient location. The Resolver itself is a collection of Java programs (components). The distribution is a self-contained package which contains all the Java libraries and other components necessary to run the Resolver.

Operation

The distribution includes a script (batch file) named *resolver.bat* which can be used to execute the

Resolver in a Windows environment [It would be easy to develop an equivalent script for a Linux or similar system; however, see *Additional Notes*.]. The *resolver.bat* script assumes it is being run from the main Resolver folder (i.e., from the folder where the distribution was unzipped). The script is intended to be invoked with a set of command line parameters, which it forwards to the actual Resolver program and which control the operation of the Resolver.

If the first parameter is the name of a file, the Resolver reads the specified file and interprets it as an *event feed file* as described above. If the first parameter is the name of a folder, the Resolver interprets it as the location of a CDP (see above), and reads its input data from that folder. Otherwise, the Resolver expects to receive command line parameters which specify how it should obtain its input.

Command line parameters other than the event feed file name must start with the characters "--" (two dashes). Each of the available command line parameters is described below. Some command line parameters require or allow additional arguments; optional arguments are shown in square brackets ([]).

Note that some command line parameters are mutually exclusive.

Command Line Options

Distributed Mode Commands

The following command line options are used to operate the Resolver in "distributed mode", where it connects to a Contest Data Server (see above) to obtain its input data. Note that these options should not be used if the first parameter to the Resolver is an event feed input file name. Note also that "--presenter" and "--connect" are mutually exclusive.

```
--presenter url user password
```

Starts a Resolver in "presenter" mode and instructs it to connect to a server at the specified URL, providing the server with the specified login and password credentials. A Resolver running in presenter mode has control over the Resolver operation, and its control commands are forwarded to the server for distribution to viewer clients. For information on user/password credentials, refer to the documentation on the Contest Data Server.

```
--connect url user password
```

Starts a Resolver in "viewer" mode, instructing it to connect to a server at the specified URL, providing the server with the specified login and password credentials. A Resolver running in viewer mode updates itself based on commands received via the server from a presenter, but has no control over the resolving process (for example, cannot "click to continue"). For information on user/password credentials, refer to the documentation on the Contest Data Server.

Control Commands

The following command line options control various aspects of the Resolver's operation; they may be used in either "stand-alone" or "distributed" mode. Note however that when running in distributed mode most options only make sense for a presenter (for example, only the presenter should specify *--fast* to control the speed of the Resolver; otherwise viewer clients will operate at a different speed than that of the presenter).

```
--info
```

Shows additional information regarding each pending run.

```
--fast [speedFactor]
```

Changes the resolving speed. The [speedFactor] option is a decimal percentage indicating the desired amount of change in the execution time for each step. Factors between 0 and 1 speed up the resolving process; factors greater than 1 slow it down. For example a speedFactor of 0.5 will double the speed (i.e., cut the time in half). If *--fast* is specified but no speedFactor is given, the default speedFactor is 0.15.

```
--medals numGold numSilver numBronze
```

Overrides the default number of medals awarded (that is, displayed by the Resolver). By default the Resolver determines the number of medals by reading information contained in the input event feed; this option allows you to specify a different number of medals in each of three categories (gold, silver, and bronze).

```
--citation "string"
```

Overrides the winning team's citation. By default the winning team is listed as "World Champion"; this option allows replacing that citation with an arbitrary string such as "First Place". The quotes are optional unless the citation contains space characters.

```
--singleStep [startRow]
```

Forces the Resolver to begin "single-stepping" (that is, requiring a key/mouse click to advance on each step of the resolving process) starting at the specified row in the standings. (For example, specifying a startRow of 10 causes single-stepping for the top 10 places in the contest.) If [startRow] is omitted then the Resolver single-steps through the entire contest.

```
--showAllFTSAwards
```

By default the Resolver displays "awards" (that is, pauses and switches to a separate "award screen") only for medalists and the winners of each "group" or "region" as defined in the input event feed. This option causes the Resolver to also switch to a separate award screen for teams whose only "award" is for being the first to solve one or more contest problems. (Note: FTS teams who also win another award—for example, a group/regional winner or a medalist—have their FTS problems noted in the citation which appears on their separate award screen.) This option may not be used if the `--ShowFTSAwardsAfterFreeze` option is specified.

```
--showFTSAwardsAfterFreeze
```

Same as the above, but only displays separate award screens for teams whose only award is being First-to-Solve a contest problem *after the scoreboard freeze*. When this option is selected, teams whose only award is that they were First-to-Solve a problem but where that solution came *before* the scoreboard was frozen will not be given a separate pause/award screen. This option may not be used if the `--ShowAllFTSAwards` option is specified.

```
--offsetMedalDisplay [numRows]
```

Normally, medalist display rows on the Resolver screen appear at the bottom of the screen. This option allows forcing the medalist display rows to begin appearing higher on the screen by some number of rows. It is used for example in the ICPC World Finals when Medalists come on stage to receive their award and remain there for the rest of the Award Ceremony; the people standing on stage can block the audience view of the bottom of the screen (and hence the next teams being processed by the Resolver). Specifying this option allows moving the medalist display rows up the screen above the heads of the people standing on stage. The default value for [numRows] if it is not specified on the command option is 4.

```
--file <file>
```

Allows loading command options from the specified <file> instead of passing them on the command line. When using this option, every command option and every optional parameter must appear on a separate line in the file.

```
--verbose
```

Shows detailed trace information on the console about what the Resolver is doing. Generally only useful for debugging purposes (use caution; it generates a **lot** of output...).

```
--help
```

Displays a help message listing the available options

Additional Notes

The Resolver is written in Java and will run on any platform supporting Java Version 7 or greater. However, it makes heavy use of screen-level graphics and is therefore heavily dependent on the graphics drivers on the platform. In our experience, Linux graphics drivers are substantially less robust than others; we have had much better success running the graphical Resolver components (standalone, presenter, and viewer clients) on Windows platforms (although we regularly run the Resolver server on Linux). Your mileage may vary (substantially).

Examples

```
resolver.bat --presenter https://169.254.80.194:8443 admin adm1n --fast 0.7 --verbose  
--ShowFTSAwardsAfterFreeze --offsetMedalDisplay 4 --info
```

The above command runs the Resolver in presenter mode, connected to a CDS at the specified URL (IP address and port) using the specified server login and password credentials, and runs reducing time by 30% (--fast 0.7), displaying verbose information on the presenter console (although not on the Resolver screen), showing Award Screens not only for group and medal winners but also for any teams whose only award was being first to solve a contest problem (but only if it was during the scoreboard freeze time as specified in the event feed received from the server). Medal awards are given for the top teams as specified in the event feed, with medal display rows offset from the bottom by 4 rows. Additional "presenter info" is displayed on the screen.

```
resolver.bat --connect https://169.254.80.194:8443 client client
```

The above command runs the Resolver in "client" mode, connecting to a CDS at the specified URL using the specified server login and password credentials. The client resolver operates under the control of a presenter-mode resolver at the same speed as the presenter and displaying only the Resolver data (no special info and no verbose mode output).

```
resolver.bat c:\eventFeed.xml --singleStep --showAllFTSAwards --citation "First Place"  
--medals 1 1 1
```

The above command runs the Resolver in "stand-alone" mode, taking its input from the specified local file. It runs in single-step mode for the entire resolving process (meaning the user must click to

advance each and every step of the resolving process). It displays award screens for first, second, and third place (one each Gold, Silver, and Bronze medal), regardless of whatever "medal count" data is contained in the event feed, and displays "First Place" as the citation for the winning team. It also displays an award screen for any team which was first to solve a problem but got no other award.

```
resolver.bat c:\contest\cdp --showFTSAwardsAfterFreeze
```

The above command runs the Resolver in "stand-alone" mode, taking its input from the specified contest data package folder. It expects the event feed data to be in a file named "contest.xml" in the specified CDP folder (c:\contest\cdp). It displays award screens for region (group) winners and for medals according to whatever "medal count" data is contained in the event feed, and also displays an award screen for any team whose only award was being the first to solve a problem after the scoreboard freeze. It includes whatever additional data it can find in the CDP (for example, team pictures and/or logos) in the output displays.

```
resolver.bat --help
```

The above command causes the Resolver to display its command parameter options on the console.