TaskTracker-tool: A Toolkit for Tracking of Code Snapshots and Activity Data During Solution of Programming Tasks

Elena Lyulina, Anastasiia Birillo, Vladimir Kovalenko, Timofey Bryksin

Introduction

Students’ coding behavior

Find gaps in understanding

Discover typical errors

Facilitate the process of teaching

Reveal code patterns

Introduction

Students' coding behavior

↓

data with interactions between a student and IDE

Find gaps in understanding

Discover typical errors

Facilitate the process of teaching

Reveal code patterns

Introduction

Students' coding behavior

data with interactions between a student and IDE

data gathering tool

Find gaps in understanding

Discover typical errors

Facilitate the process of teaching

Reveal code patterns

Existing Tools

Tool	Target IDE	Tracking data	Description
ClockIt	BlueJ 	IDE actions	A visualization tool to compare behavior of users with different programming experience
Marmoset plugin	Eclipse 	**coarse-grained** seq. snapshots of code	A testing system to grade submissions of problems with VCS synchronisation
DevEvent Tracker	Eclipse 	**fine-grained** seq. snapshots of code	A part of the Web-CAT system to analyze solution process with many features available
Blackbox	BlueJ 	**fine-grained** seq. snapshots of code + IDE actions	A data collection project, including large open dataset with actions and code from various users

Table 1: Existing Tools

Existing Tools

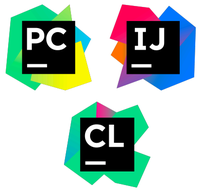
Tool	Target IDE	Tracking data	Description
ClockIt	BlueJ 	IDE actions	A visualization tool to compare behavior of users with different programming experience
Marmoset plugin	Eclipse 	**coarse-grained** seq. snapshots of code	A testing system to grade submissions of problems with VCS synchronisation
DevEvent Tracker	Eclipse 	**fine-grained** seq. snapshots of code	A part of the Web-CAT system to analyze solution process with many features available
Blackbox	BlueJ 	**fine-grained** seq. snapshots of code + IDE actions	A data collection project, including large open dataset with actions and code from various users
TaskTracker	IntelliJ 	**fine-grained** seq. snapshots of code + IDE actions	A flexible tool to collect and analyze task-specific data with every code change tracked

Table 1: Existing Tools

Existing Tools

Tool	Target IDE	Tracking data	Description
ClockIt	BlueJ 	IDE actions	A visualization tool to compare behavior of users with different programming experience
Marmoset plugin	Eclipse 	**coarse-grained** seq. snapshots of code	A testing system to grade submissions of problems with VCS synchronisation
DevEvent Tracker	Eclipse 	**fine-grained** seq. snapshots of code	A part of the Web-CAT system to analyze solution process with many features available
Blackbox	BlueJ 	**fine-grained** seq. snapshots of code + IDE actions	A data collection project, including large open dataset with actions and code from various users

Table 1: Existing Tools

Our Contributions

Plugin to track code changes, working in conjunction with *ActivityTracker* plugin

Server to facilitate data gathering process

Data post-processing tool to analyze collected data

TaskTracker-tool

Public **dataset** of problem solving activity data of 148 students

TaskTracker-tool: plugin

- **User interface:**
  - filling a survey
  - choosing task to solve
  - viewing task description

- **Data collection:**
  - using *ActivityTracker*[1] plugin to track actions
  - tracking **only** task-related files
  - taking snapshots of **every** code change
  - saving data in `.csv` format

[1]**ActivityTracker**: https://github.com/dkandalov/activity-tracker

TaskTracker-tool: server

Information common for all languages is filled once

Otherwise, it should be filled for each language

```
{
    'key': 'pies',
    'examples':
        [
            { 'input': '10\n15\n2', 'output': '20 30'},
            { 'input': '2\n50\n4', 'output': '10 0'},
            { 'input': '2\n50\n0', 'output': '0 0' }
        ],
    'descriptions':
        [
            {
                'language': 'en',
                'info': {
                    'name': 'Pies',
                    'description': 'A single pie costs A dollars and B cents in the cafe. ' +
                        'Calculate how many dollars and cents you would need to buy N pies.',
                    'input': 'The program receives three numbers as an input:\n' +
                        'A - how many dollars a pie costs;\n' +
                        'B - how many cents a pie costs;\n' +
                        'N - how many pies do you need to buy.',
                    'output': 'Print out two numbers: the cost of N pies in dollars and cents.'
                }
            }
        ]
},
```

Figure 1: The tasks config file

TaskTracker-tool: server

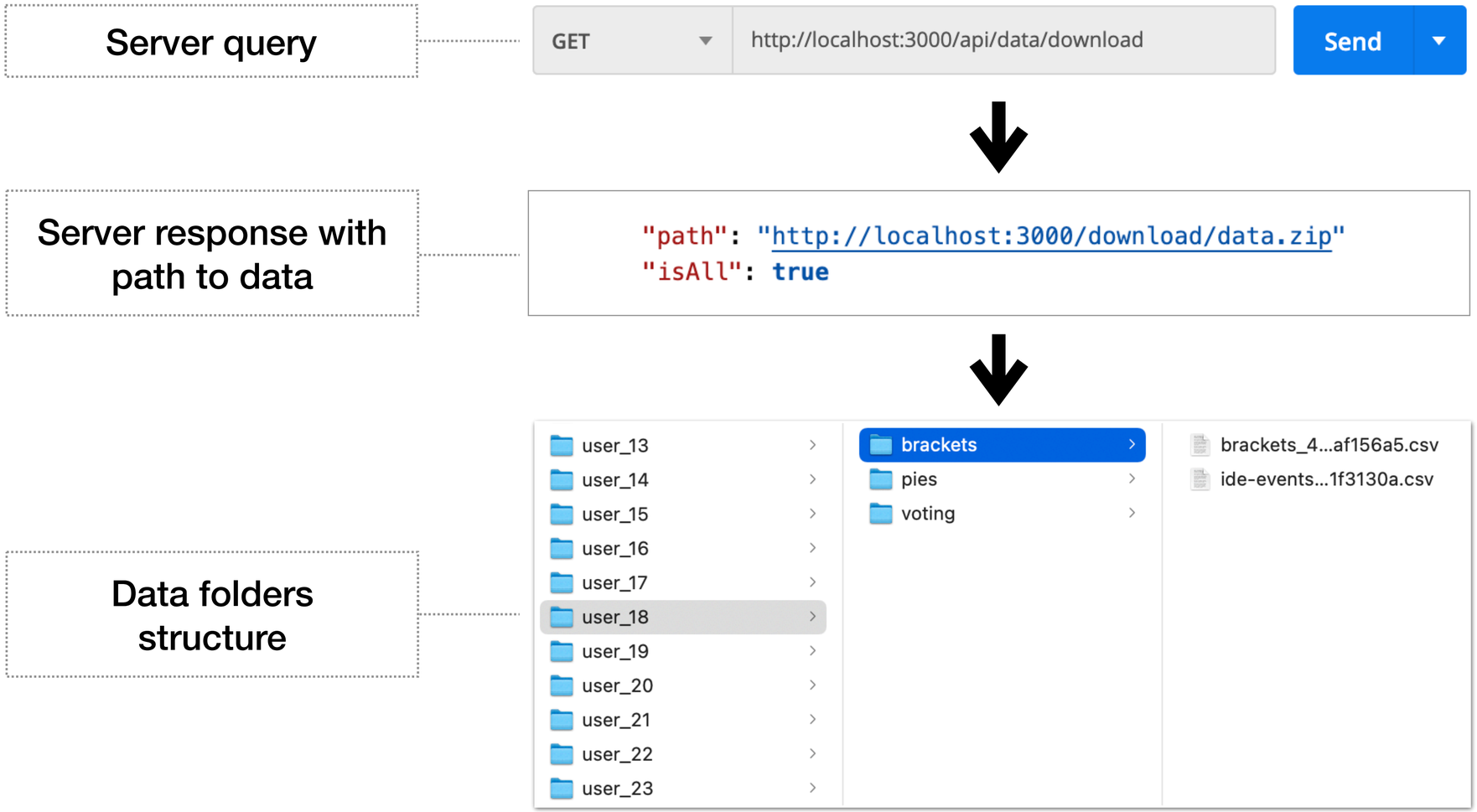


Figure 2: Data downloading pipeline

TaskTracker-tool: data post-processing

	date	fileName	fragment	age
1	2019-12-20T13:45:07.784+03:00	pies.py	a	17.0
2	2019-12-20T13:45:07.869+03:00	pies.py	a =	17.0
3	2019-12-20T13:45:08.085+03:00	pies.py	a =	17.0
4	2019-12-20T13:45:08.126+03:00	pies.py	a = i	17.0
5	2019-12-20T13:45:08.500+03:00	pies.py	a = in	17.0
6	2019-12-20T13:45:08.502+03:00	pies.py	a = in(	17.0
7	2019-12-20T13:45:08.778+03:00	pies.py	a = in()	17.0
8	2019-12-20T13:45:08.779+03:00	pies.py	a = in)	17.0
9	2019-12-20T13:45:08.924+03:00	pies.py	a = in	17.0
10	2019-12-20T13:45:09.049+03:00	pies.py	a = int	17.0
11	2019-12-20T13:45:09.051+03:00	pies.py	a = int(	17.0
12	2019-12-20T13:45:09.529+03:00	pies.py	a = int()	17.0
13	2019-12-20T13:45:09.571+03:00	pies.py	a = int(i)	17.0
14	2019-12-20T13:45:09.795+03:00	pies.py	a = int(in)	17.0
15	2019-12-20T13:45:09.870+03:00	pies.py	a = int(inp)	17.0
16	2019-12-20T13:45:09.946+03:00	pies.py	a = int(inpu)	17.0
17	2019-12-20T13:45:10.130+03:00	pies.py	a = int(input)	17.0
18	2019-12-20T13:45:10.132+03:00	pies.py	a = int(input()	17.0

Figure 3: TaskTracker data format

TaskTracker-tool: data post-processing

- Merging *ActivityTracker* and *TaskTracker* data:

	at_date	action
1	13:45:10.131	EditorRight
2	13:45:10.920	EditorEnter

+

	date	fragment
1	13:45:09.946	a = int(inpu)
2	13:45:10.130	a = int(input)
3	13:45:10.132	a = int(input()
4	13:45:10.917	a = int(input())
5	13:45:10.923	a = int(input())⏎
6	13:45:11.366	a = int(input())⏎b

a) ActivityTracker and TaskTracker files for a task

→

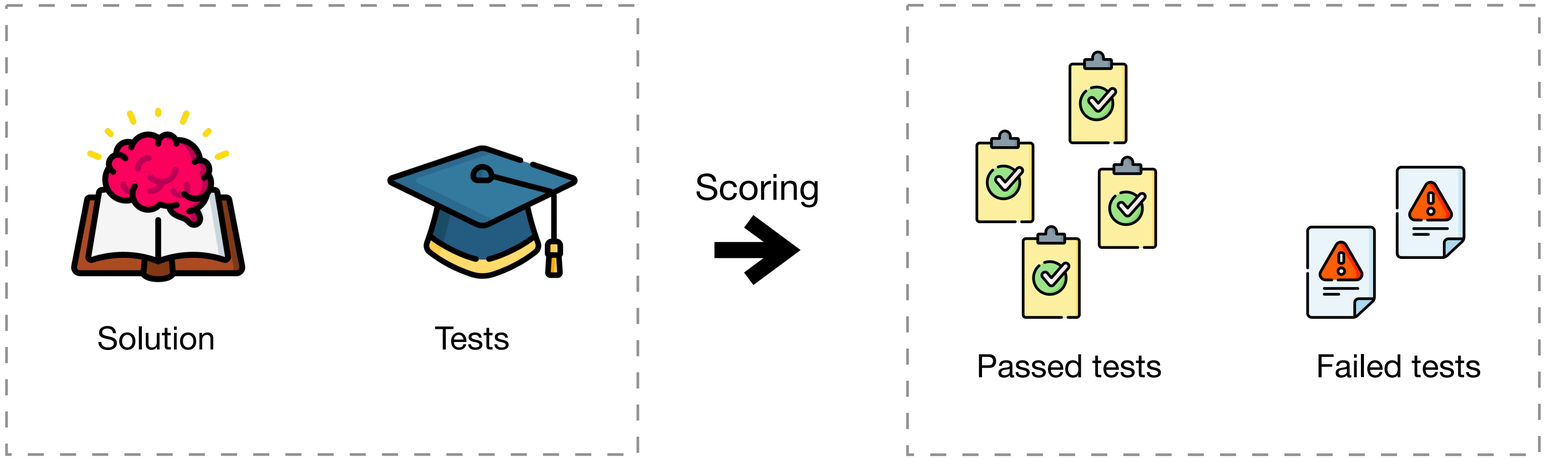
	date	fragment	at_date	action
1	13:45:09.946	a = int(inpu)	*<null>*	*<null>*
2	13:45:10.130	a = int(input)	13:45:10.131	EditorRi…
3	13:45:10.132	a = int(input()	*<null>*	*<null>*
4	13:45:10.917	a = int(input())	13:45:10.920	EditorEn…
5	13:45:10.923	a = int(input())⏎	*<null>*	*<null>*
6	13:45:11.366	a = int(input())⏎b	*<null>*	*<null>*

b) Merged ActivityTracker and TaskTracker files

Figure 4: Merging *ActivityTracker* and *TaskTracker* data pipeline

TaskTracker-tool: data post-processing

- Merging *ActivityTracker* and *TaskTracker* data;
- Scoring solutions:



$$score = \frac{passed\ \ tests}{all\ \ tests}$$

Figure 5: Scoring solutions pipeline

TaskTracker-tool: data post-processing

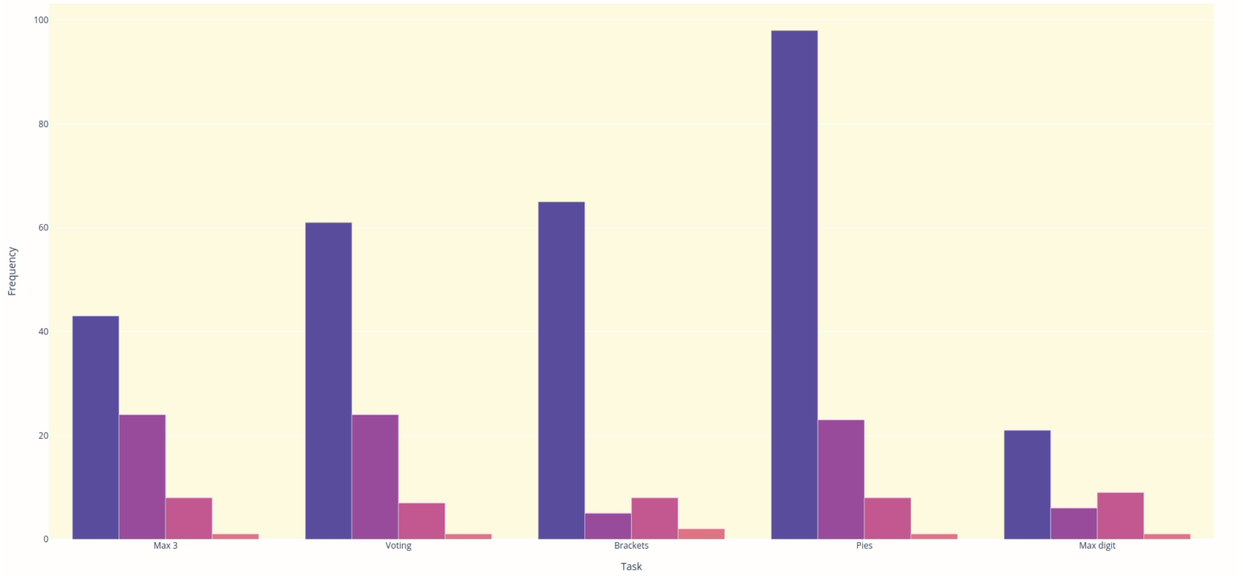
- Merging *ActivityTracker* and *TaskTracker* data;
- Scoring solutions;
- Removing intermediate code changes:

	fragment
1	a = input
2	a = input(
3	a = input()
4	a = input()⏎
5	a = input()⏎b
6	a = input()⏎b =
7	a = input()⏎b =
8	a = input()⏎b = 5
9	a = input()⏎b = 5⏎

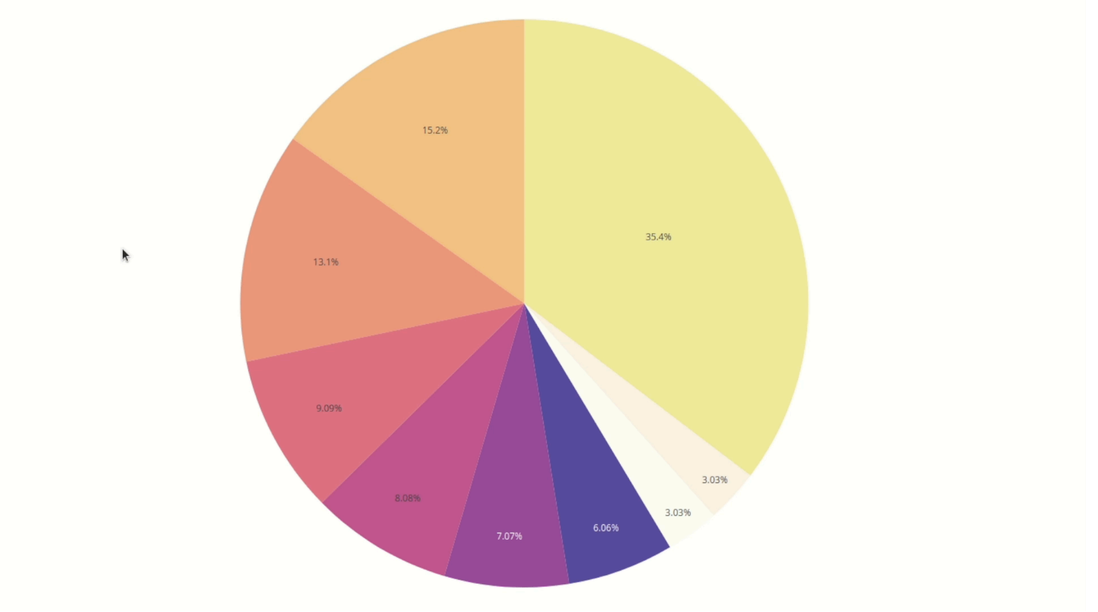
	fragment
1	a = input
2	a = input()⏎
3	a = input()⏎b = 5⏎

Figure 6: Removing intermediate code changes pipeline

TaskTracker-tool: data post-processing



a) Tasks and Languages distribution: bar chart



b) Participants' age distribution: pie chart

Figure 7: Statistics charts

TaskTracker-tool: use cases

- **Programming courses:**

Observed problem-solving in class

Course improvement

Personalized help

Cheating detection

- **Data gathering:**

Suitable for remote gathering

Works with any UI language

Easy to configure, install and use

With care for users' privacy

Dataset: description

Task	Description
Pies	A single pie costs A dollars and B cents in the cafe. Calculate how many dollars and cents one needs to pay for N pies
Max of 3	Print the largest of three numbers in the input
Zero	Check if there are zeros among numbers in the input
Voting	Given three numbers, each of them being 1 or 0, determine which one occurs more often: 1 or 0. Print the number that occurs more often
Max digit	Given a string containing only digits, find and print the largest digit
Brackets	Place opening and closing brackets into the input string like this: for odd length: `example` →`e(x(a(m)p)l)e`; for even length: `card` → `c(ar)d`, but not `c(a()r)d`.

Table 2: Task descriptions

Dataset: statistics

- **148** participants;
- **11** to **40** years old (mean age is **19**);
- **4** languages: Python, Java, Kotlin, or C++;
- collected:
  - **326** correct solutions,
  - **148** incorrect solutions;
- anonymized and open.

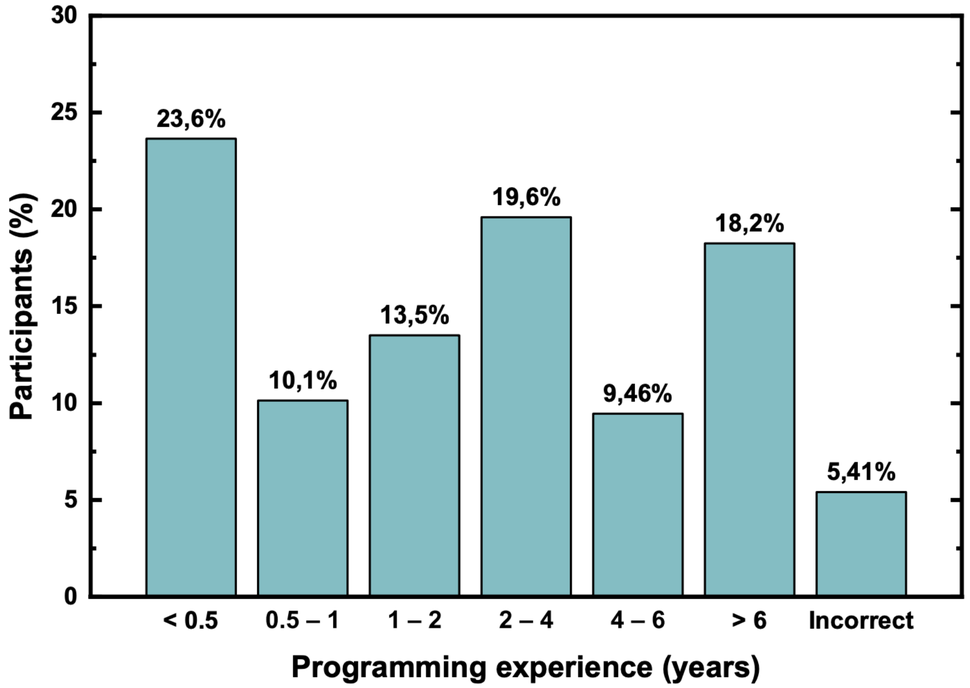


Figure 8: Distribution of participants' experience

Dataset: analysis

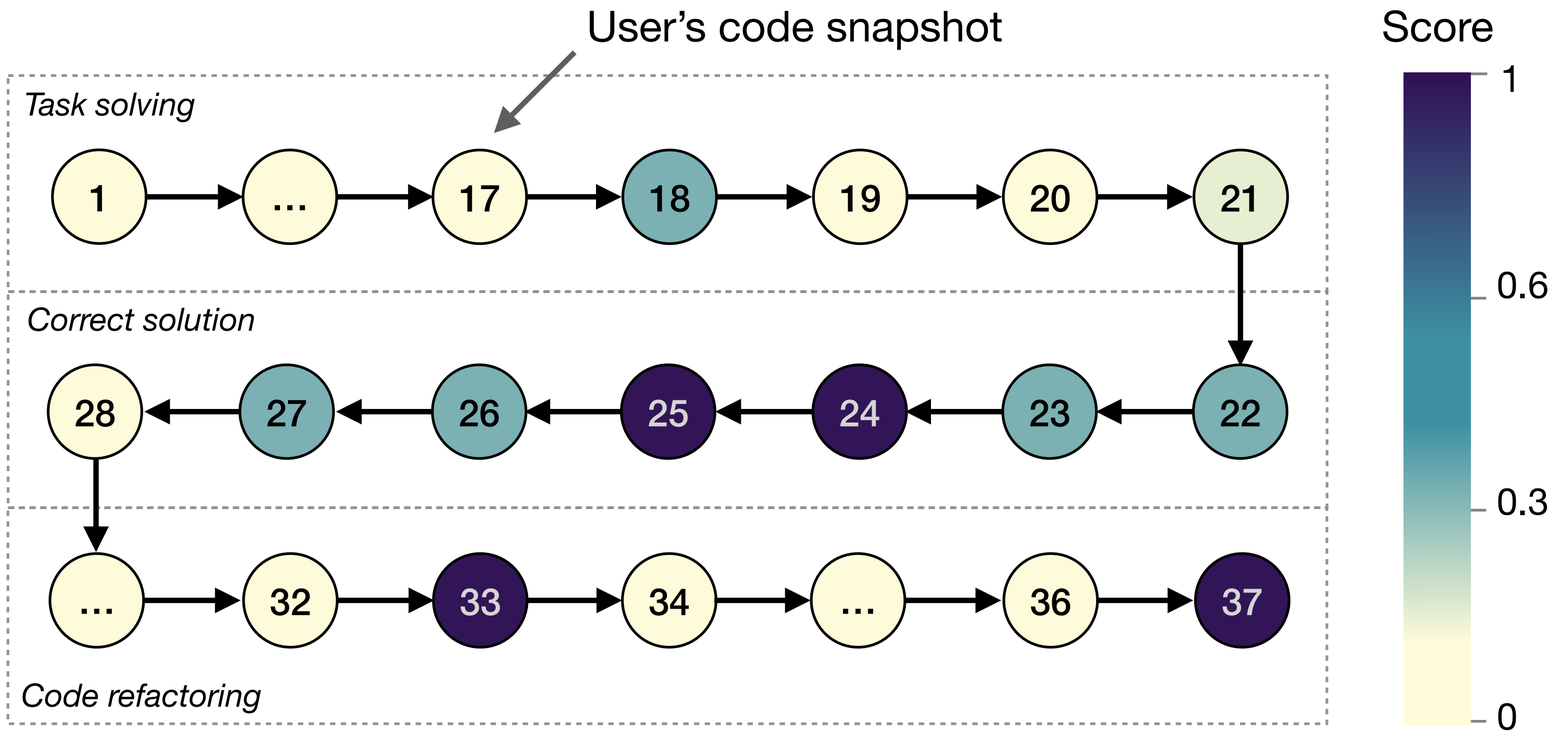


Figure 9: Changes of task score during a sample solution

Dataset: analysis

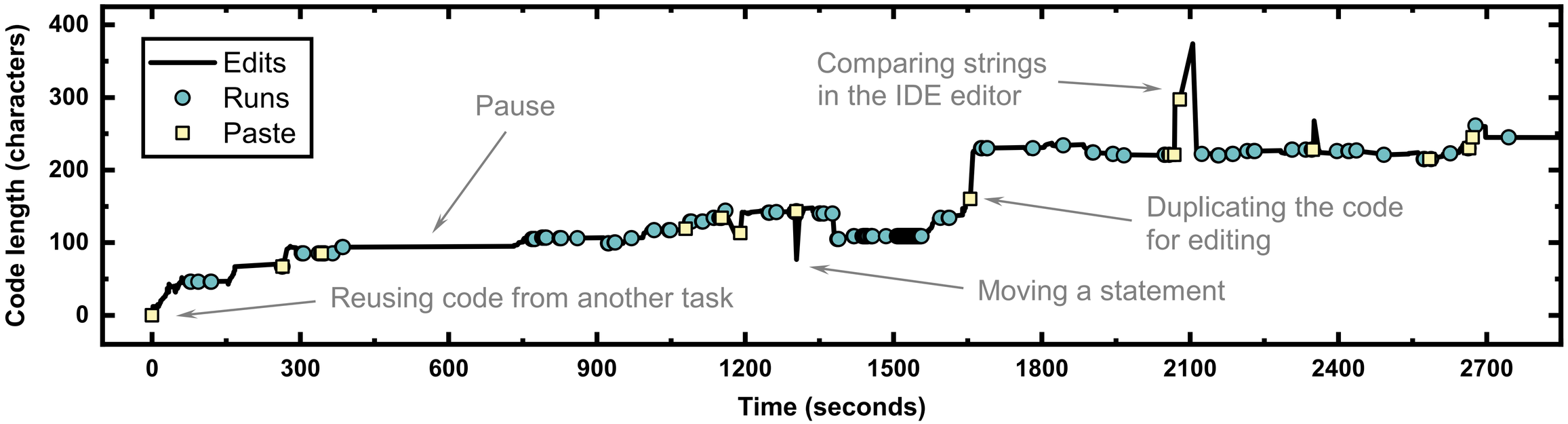


Figure 10: Actions performed in IDE during solution of a task

Dataset: use cases

Experience and feature use

Influence of age on feature use

Actions after solution

Common errors

Advanced solution metrics

Generating personalized hints

Conclusion

- **TaskTracker-tool**
  - gather task-specific data of solution activity
  - collect data on server
  - perform basic analysis
- Public **dataset** of 148 students

Thank you!

- **Elena Lyulina** - elena.lyulina@jetbrains.com

- **Links**:

Plugin

Server

Dataset

Post-processing tool

Dataset anonymizer