# Exploring Student Misconceptions about Concurrency via the Domain-Specific Programming Language Sonic Pi

Università di Genova

Giorgio Delzanno, Giovanna Guerrini, Daniele Traversaro
DIBRIS, Università degli Studi di Genova, Italy
{giorgio.delzanno,giovanna.guerrini,daniele.traversaro}@dibris.unige.it

## Background and motivations

- Thanks to multicore architectures, the importance of **concurrent** and **multi-threaded programming** continues to grow.
- For this reason, many universities have incorporated these concepts into their **introductory** programming courses.
- Concurrent programming, however, is still considered a difficult topic for several students.
- Domain-specific languages such as **Sonic Pi**, designed for **live music coding**, provide simplified programming constructs and immediate feedback for concepts related to concurrency.

## Proposal

- Our paper investigates the combined use of **Sonic Pi** and **Team-Based Learning** (TBL) to mitigate the difficulties in **early exposure to concurrency**.
- Our **research goal** is to explore whether the use of Sonic Pi can **support students**, especially in the **early stages**, to understand concurrent programming, by **playing and hearing common errors and misconceptions**.

## Sonic Pi approach

We reformulate **misconceptions** in Sonic Pi:

- **Sequential** versus **interleaving** execution of multiple threads: **playing a single instrument** or **multiple instruments simultaneously**.
- Threads execution out of control: **out of synch** or **drifting effect**.
- Confusion between **function-** and **thread-local variables**: **wrong use of data structures** used to code **music patterns**.
- Data races: **chords** versus **random note sequences**.
- A program **sounds** correct after testing only a subset of all possible interleavings.

## TBL approach

We use the following TBL lecture plan:

- Pre-class individual study;
- Readiness Assurance Process: **iRAT**, **tRAT**, and **clarifications**;
- Team App (**tAPP**) on common errors and misconceptions;
- Gallery walk;
- Final individual questionnaire

## Teaching experiments

Teaching experiments with students of the **Computer Science Bachelor of our University**, involving **184 participants**:

- **130** students of **Computer Architectures (CA)**, a first year undergraduate course;
- **54** students of **Concurrent and Distributed Algorithms (CDA)**, a third year undergraduate course.

## Example of iRAT quiz

| Consider the following code: | What happens if you run it? |
|---|---|
| `live_loop :foo do` `sample :ambi_choir` `sleep 0.5` `end` `in_thread do` `sample :ambi_drone` `end` | • The sample in the `live loop` command is played repeatedly, while the sample in the `in_thread` command is played only once.<br>• It is not possible to execute both a thread and a live loop concurrently (runtime error).<br>• Both the live loop and the thread sample are played infinitely.<br>• Only the live loop will be executed. |

## Example of tAPP task

```
use_synth :piano
notes=[:E3,:G3,:B3]
counter=0

define :foo do |x|
  in_thread do
    play notes[x]
  end
end

3.times do
  foo counter
  counter+=1
end
```

**Voting Cards**

Which statement is true?

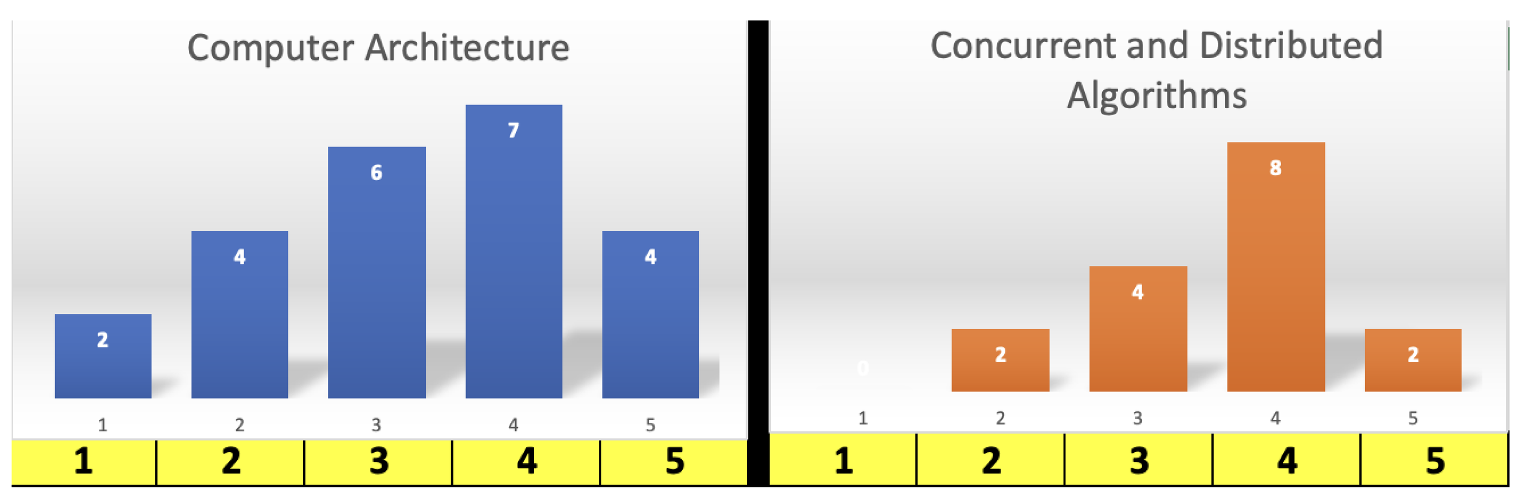
The program has no race conditions.

The program is single threaded.

The auditory output is a sequence of three notes.

The auditory output is the E minor chord.

## Is Sonic Pi useful to understand concurrency?




## References

[1] S. Aaron, A.F. Blackwell, P. Burnard. The development of Sonic Pi and its use in educational partnerships: Co-creating pedagogies for learning computer programming. *Journal of Music Technology and Education*, 9.1 (2016): 75-94.

[2] F. Strömbäck. (2023). Teaching and Learning Concurrent Programming in the Shared Memory Model. PhD thesis, Linköping University.