Fair Join Pattern Matching for Actors (Artifact)

Philipp Haller
KTH Royal Institute of Technology, Stockholm, Sweden

Ayman Hussein
Technical University of Denmark, Lyngby, Denmark

Hernán Melgratti
University of Buenos Aires & Conicet, Argentina

Alceste Scalas
Technical University of Denmark, Lyngby, Denmark

Emilio Tuosto
Gran Sasso Science Institute, L'Aquila, Italy

Abstract

Join patterns provide a promising approach to the development of concurrent and distributed message-passing applications. Several variations and implementations have been presented in the literature — but various aspects remain under-explored: in particular, how to specify a suitable notion of message matching, how to implement it correctly and efficiently, and how to systematically evaluate the implementation performance.

In this work we focus on actor-based programming, and study the application of join patterns with conditional guards (i.e., the most expressive and challenging version of join patterns in literature). We formalise a novel specification of *fair and deterministic join pattern matching*, ensuring that older messages are always consumed if they can be matched. We present a *stateful, tree-based join pattern matching algorithm* and prove that it correctly implements our fair and deterministic matching specification. We present a novel Scala 3 actor library (called `JoinActors`) that implements our join pattern formalisation, leveraging macros to provide an intuitive API. Finally, we evaluate the performance of our implementation, by introducing a systematic benchmarking approach that takes into account the nuances of join pattern matching (in particular, its sensitivity to input traffic and complexity of patterns and guards).

2012 ACM Subject Classification Software and its engineering → Formal language definitions; Software and its engineering → Domain specific languages; Software and its engineering → Concurrent programming languages; Software and its engineering → Distributed programming languages; Theory of computation → Process calculi

Keywords and phrases Concurrency, join patterns, join calculus, actor model

Digital Object Identifier 10.4230/DARTS.10.2.8

Funding *Ayman Hussein*: Research supported by the Horizon Europe grant 101093006 (TaRDIS).
Alceste Scalas: Research partly supported by the Horizon Europe grant 101093006 (TaRDIS).
Emilio Tuosto: Research partly supported by the EU H2020 RISE programme under the Marie Skłodowska-Curie grant agreement No 778233, the PRIN PNRR project DeLICE (P20223T2MF), "by the MUR dipartimento di eccellenza", and by PNRR MUR project VITALITY (ECS00000041), Spoke 2 ASTRA – Advanced Space Technologies and Research Alliance.

Acknowledgements This work was inspired by the group discussion on "Join patterns / synchronisation - the next generation" [1, page 54] at the Dagstuhl Seminar 21372; we thank the organisers of the meeting and Schloss Dagstuhl — Leibniz Center for Informatics for making this work possible.
We thank Omar Inverso for the technical support he provided for our experimental evaluation, Roland Kuhn for fruitful discussions on the shop floor use case, António Ravara for some useful suggestions, and Antoine Sébert for an implementation of join patterns using Scala 3 macros [2].
We thank the anonymous reviewers for their comments and suggestions.

Dagstuhl Artifacts Series
Schloss Dagstuhl – Leibniz-Zentrum für Informatik,
Dagstuhl Publishing, Germany

Related Article Philipp Haller, Ayman Hussein, Hernán Melgratti, Alceste Scalas, and Emilio Tuosto, "Fair Join Pattern Matching for Actors", in 38th European Conference on Object-Oriented Programming (ECOOP 2024), LIPIcs, Vol. 313, pp. 17:1–17:28, 2024.
https://doi.org/10.4230/LIPIcs.ECOOP.2024.17

Related Conference 38th European Conference on Object-Oriented Programming (ECOOP 2024), September 16–20, 2024, Vienna, Austria

Evaluation Policy The artifact has been evaluated as described in the ECOOP 2024 Call for Artifacts and the ACM Artifact Review and Badging Policy.

1 Scope

This artifact supports the Fair Join Pattern Matching for Actors theory and experiments discussed in the accompanying paper. We have implemented a Scala 3 library, `JoinActors`, for actors with join patterns, featuring both "brute-force" and tree-based stateful implementations of our deterministic fair matching semantics. Our library can be used with the off-the-shelf Scala 3 compiler, and leverages macros to provide an intuitive API.

2 Content

The artifact contains the following:

- `join-actors`: The source code for the `JoinActors` library implementation (including the benchmarks). See the README file in the directory for more information.
- `experiment-results`: Contains python scripts to reproduce the figures in the paper. See the README file in the directory for more information.
- `evrete-smarthouse`: Contains the source code for the Evrete[1]- based implementation of the Smart House example. See the README file in the directory for more information.

3 Getting the artifact

The artifact endorsed by the Artifact Evaluation Committee is available free of charge on the Dagstuhl Research Online Publication Server (DROPS). In addition, the library is also available at: `https://github.com/a-y-man/join-actors`.

4 Tested platforms

OS and resource (CPU, memory, disk, GPU) used by the authors for evaluation is:

- A dual Xeon E5-2687W 8-core 3.10GHz processor and 128GB of memory, running 64-bit Linux 5.10.27.

The artifact has also been tested on the following platform:

- An 11th Gen Intel Core i7-1185G7 @ 8x 4.8GHz and 32GB of memory, running 64-bit Linux 6.8.0.

5 License

The artifact is available under license Apache License 2.0.

[1] `https://www.evrete.org`

6 MD5 sum of the artifact

5640c68ae6bba8633bc30d0f01cbb87d

7 Size of the artifact

22.9 MiB

References

1 Mariangiola Dezani, Roland Kuhn, Sam Lindley, and Alceste Scalas. Behavioural Types: Bridging Theory and Practice (Dagstuhl Seminar 21372). *Dagstuhl Reports*, 11(8):52–75, 2022. doi:10.4230/DagRep.11.8.52.

2 Antoine Louis Thibaut Sébert. Join-patterns for the actor model in scala 3 using macros. Master's thesis, DTU Department of Applied Mathematics and Computer Science, 2022. Available at https://findit.dtu.dk/en/catalog/62f83d3680aa6403a4ccc0ab.