The Meta-CSP Framework

A Java API for meta-constraint reasoning

Federico Pecora
Center for Applied Autonomous Sensor Systems, Örebro Unievrsity
federico.pecora@oru.se

Website:	http://metacsp.org
GitHub repo:	https://github.com/FedericoPecora/meta-csp-framework
Stable release:	1.3.5

1 What is meta-constraint reasoning?

Constraint Satisfaction Problems (CSP) consist of a finite set of variables, each associated with a finite domain, and a set of constraints which restrict simultaneous assignments to variables. A Meta-CSP is CSP formulation of a combinatorial problem which builds on lower-level CSPs.

2 What does this API do?

This software framework provides tools for developing solvers for problems that can be cast as Meta-CSPs. The framework includes several built-in CSP and Meta-CSP problem solvers which can be used as "ingredients" for defining more sophisticated solvers. Among these, hybrid problem solvers which exemplify the natural predisposition of Meta-CSPs for solving hybrid reasoning problems.

3 Documentation

Developers can refer to the API documentation (Javadoc) for reference, and to the examples in the following packages for getting started:

- Package `org.metacsp.examples` contains examples showcasing the use of "basic" constraint-based solvers. These include the STP solver `APSPSolver`, the Boolean SAT solver `BooleanSatisfiabilitySolver`, and spatial reasoning engines for DE9IM and RCC-5/8. These solvers are used in other examples (see below) to build hybrid problem solvers using the Meta-CSP methodology.
- Package `org.metacsp.examples.multi` contains examples of how several "basic" solvers can be composed in a loosely-coupled way, that is, where decision made by one solver do not change the search space of another solver. Examples include the `SpatioTemporalVariableSolver`, which joins an STP solver with a DE9IM solver, the `SymbolicVariableConstraintSolver`, which exploits Boolean SAT to obtain a simple state variable solver.
- Package `org.metacsp.examples.meta` contains examples that showcase how the concept of meta-variable and meta-constraint is used to determine how decisions made by different solvers affect each others' search space. Examples showcase a `ReusableResourceScheduler`, a `StateVariableScheduler`, a timeline-based planner called `SimplePlanner`, and an integrated context inference and planning framework called SAM [10] used in several EU projets [9, 2, 4] (see class `TestProactivePlanningAndDispatching`).

4 Using the framework as a library

The Meta-CSP Framework is available on Maven Central. The Maven dependency declaration is:

```
<dependency>
  <groupId>org.metacsp</groupId>
  <artifactId>meta-csp-framework</artifactId>
  <version>1.3.5</version>
</dependency>
```

The Gradle dependency declaration is:

```
compile 'org.metacsp:meta-csp-framework:1.3.5'
```

Please note that the current stable release may have changed (it was 1.3.5 at the time of writing).

Alternatively, the latest binary, source and Javadoc JARs are available from the Maven Central website[1].

5 Compiling from source

The framework as well as the API documentation (Javadoc) can be built through Gradle. A Gradle redistributable is included in the master branch, so there is no need to install Gradle on your machine.

5.1 Gradle build instructions

Enter the directory with the file build.gradle and issue the command:

```
$ ./gradlew install    #(on Unix-based systems)
$ gradlew.bat install  #(on Windows-based systems)
```

To test the build, issue the following:

```
$ ./gradlew run        #(on Unix-based systems)
$ gradlew.bat run      #(on Windows-based systems)
```

The `clean` target will clean up the build directory. The target `javadoc` can be used to generate the API documentation (Javadoc), which will be placed in `build/docs/javadoc`.

5.2 Preparing an Eclipse project

If developing in Eclipse, consider using the eclipse target:

```
$ ./gradlew eclipse    #(on Unix-based systems)
$ gradlew.bat eclipse  #(on Windows-based systems)
```

This will prepare the directory with `.classpath`, `.settings` and `.project` files. The directory can then be used as source for a new Eclipse project which will have all dependencies properly set.

5.3 Using the compiled library in other Gradle projects

The install target builds the artifact `meta-csp-framework-0.0.0-SNAPSHOT.jar`, and places it into your local Maven repository (the location is `~/.m2/repository/org/metacsp/meta-csp-framework/` on a Unix-based system). You can use the newly compiled version of the framework in another Gradle-based project by including the following in the project's `build.gradle`:

```
repositories {
   mavenLocal()
   //any other repo you may need for your project
}

dependencies {
   compile 'org.metacsp:meta-csp-framework:0.0.0-SNAPSHOT'
   //any other dependency you may have
}
```

6 Contributors and applications

The Meta-CSP Framework is developed by Federico Pecora, with significant contributions from the open source community. Contributions are also provided by Scania Research and Development, which currently uses the framework for developing fleet optimization solutions. The HTN-based [3] planner CHIMP [15, 14] is implemented using the Meta-CSP Framework.

[1] Please insert `g:org.metacsp AND a:meta-csp-framework` in the search on `http://search.maven.org`.

The framework provides an implementation of Trajectory Envelopes, a spatio-temporal representation of robot paths that is used for trajectory scheduling and multi-robot coordintaion [11, 1] . This has been used in applications ranging from mining [7, 8, 5] to warehouse automation [6]. The `coordination_oru` framework for integrated motion planning, coordination and control [13] uses the Trajectory Envelope representation, and the corresponding implementation [12] uses the Meta-CSP Framework as a library.

7 Sponsors

This project is supported by

- The Semantic Robots Research Profile, funded by the Swedish Knowledge Foundation, see `http://semanticrobots.oru.se` for details.
- The ILIAD Project, funded by the EC H2020 Program, see `https://iliad-project.eu` for details.
- Örebro University
- The RACE (Robustness by Autonomous Competence Enhancement) Project, funded by the EC FP7 Program, see `http://project-race.eu/` for details.

8 License

References

[1] H. Andreasson, A. Bouguerra, M. Cirillo, D. Dimitrov, D. Driankov, L. Karlsson, A. Lilienthal, F. Pecora, J.P. Saarinen, A. Sherikov, and T. Stoyanov. Autonomous transport vehicles: where we are and what is missing. *IEEE Robotics & Automation Magazine*, 22(1):64–75, 2015.

[2] Filippo Cavallo, Raffaele Limosani, Alessandro Manzi, Manuele Bonaccorsi, Raffaele Esposito, Maurizio Di Rocco, Federico Pecora, Giancarlo Teti, Alessandro Saffiotti, and Paolo Dario. Development of a socially believable multi-robot solution from town to home. *Cognitive Computation*, 6(4):954–967, 2014.

[3] Ilche Georgievski and Marco Aiello. Htn planning: Overview, comparison, and beyond. *Artificial Intelligence*, 222:124–156, 2015.

[4] Joachim Hertzberg, Jianwei Zhang, Liwei Zhang, Sebastian Rockel, Bernd Neumann, Jos Lehmann, Krishna Sandeep Reddy Dubba, Anthony G Cohn, Alessandro Saffiotti, Federico Pecora, et al. The race project: Robustness by autonomous competence enhancement. *KI-Künstliche Intelligenz*, 28(4):297–304, 2014.

[5] Masoumeh Mansouri, Henrik Andreasson, and Federico Pecora. Hybrid reasoning for multi-robot drill planning in open-pit mines. *Acta Polytechnica*, 56(1):47–56, 2016.

[6] Masoumeh Mansouri, Henrik Andreasson, and Fredrico Pecora. Towards hybrid reasoning for automated industrial fleet management. In *Workshop on Hybrid Reasoning at IJCAI'15, Buenos Aires, Argentina, 25-31 July, 2015*, 2015.

[7] Masoumeh Mansouri, Bruno Lacerda, Nick Hawes, and Federico Pecora. Multi-robot planning under uncertain travel times and safety constraints. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2019. (to appear).

[8] Masoumeh Mansouri, Fabien Lagriffoul, and Federico Pecora. Multi vehicle routing problem with nonholonomic constraints and dense dynamic obstacles. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017.

[9] Masoumeh Mansouri and Federico Pecora. A robot sets a table: a case for hybrid reasoning with different types of knowledge. *Journal of Experimental & Theoretical Artificial Intelligence*, 28(5):801–821, 2016.

[10] F. Pecora, M. Cirillo, F. Dell'Osa, J. Ullberg, and A. Saffiotti. A constraint-based approach for proactive, context-aware human support. *Journal of Ambient Intelligence and Smart Environments*, 4(4):347–367, 2012.

[11] F. Pecora, M. Cirillo, and D. Dimitrov. On mission-dependent coordination of multiple vehicles under spatial and temporal constraints. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[12] Federico Pecora. *An Online Multi-Robot Coordination Algorithm based on Trajectory Envelopes*, 2017. `http://github.com/FedericoPecora/coordination_oru`.

[13] Federico Pecora, Henrik Andreasson, Masoumeh Mansouri, and Vilian Petkov. A loosely-coupled approach for multi-robot coordination, motion planning and control. In *Proc. of the International Conference on Automated Planning and Scheduling (ICAPS)*, 2018.

[14] Sebastian Stock, Masoumeh Mansouri, Federico Pecora, and Joachim Hertzberg. Hierarchical hybrid planning in a mobile service robot. In *Joint German/Austrian Conference on Artificial Intelligence (Künstliche Intelligenz)*, pages 309–315. Springer International Publishing, 2015.

[15] Sebastian Stock, Masoumeh Mansouri, Federico Pecora, and Joachim Hertzberg. Online task merging with a hierarchical hybrid task planner for mobile service robots. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 6459–6464. IEEE, 2015.