Hipster4j

A powerful and friendly heuristic search library implemented in Java.

What's Hipster4j?

The aim of Hipster4j is to provide an easy to use yet powerful and flexible type-safe Java library for heuristic search. Hipster relies on a flexible model with generic operators that allow you to reuse and change the behavior of the algorithms very easily. Algorithms are also implemented in an iterative way, avoiding recursion. This has many benefits: full control over the search, access to the internals at runtime or a better and clear scale-out for large search spaces using the heap memory.

You can use Hipster4j to solve from simple graph search problems to more advanced state-space search problems where the state space is complex and weights are not just double values but custom defined costs.

Features

The current version of the library comes with some very well-known and wide used search algorithms. We're working to add more algorithms soon:

- Search algorithms:
  - Uninformed search:
    - DFS: Depth-First-Search.
    - BFS: Breadth-First-Search.
    - Dijkstra's algorithm.
    - Bellman-Ford.
  - Informed search:
    - A star (A*).
    - IDA star (IDA*), Iterative Deepening A*.
    - AD star (AD*): Anytime Dynamic A*.
  - Local search:
    - Hill-Climbing.
    - Enforced-Hill-Climbing.
  - Multiobjective search
    - Multiobjective LS algorithm. Original paper: Martins, E. D. Q. V., & Santos, J. L. E. (1999). "The labeling algorithm for the multiobjective shortest path problem". Departamento de Matematica, Universidade de Coimbra, Portugal, Tech. Rep. TR-99/005 (see an example)
- 3rd party adapters:
  - Java Universal/Graph (JUNG) adapter.

If you don't find the algorithm or the feature you are looking for, please consider contributing to Hipster!. You can open a new issue or better fork this repository and create a pull request with your contribution.

Getting started

The easiest way to use Hipster is adding it as a dependency with your favourite dependency manager. Maven users can include the library using the following snippet:

Snapshots

You can use the latest (unstable) version of Hipster under development. Just add the following dependency into your pom.xml:

```
<!-- Use sonatype oss public for snapshots -->
<repositories>
 <repository>
  <id>sonatype-oss-public</id>
  <url>https://oss.sonatype.org/content/groups/public/</url>
  <snapshots>
   <enabled>true</enabled>
  </snapshots>
 </repository>
</repositories>

<dependencies>
 <!--
  Add this dependency under your pom.xml <dependencies> section to add
  all the dependencies of Hipster to your project. Add hipster-core
  instead of hipster-all for basic functionality.
 -->
 <dependency>
  <groupId>es.usc.citius.hipster</groupId>
  <artifactId>hipster-all</artifactId>
  <version>1.0.2-SNAPSHOT</version>
 </dependency>
</dependencies>
```

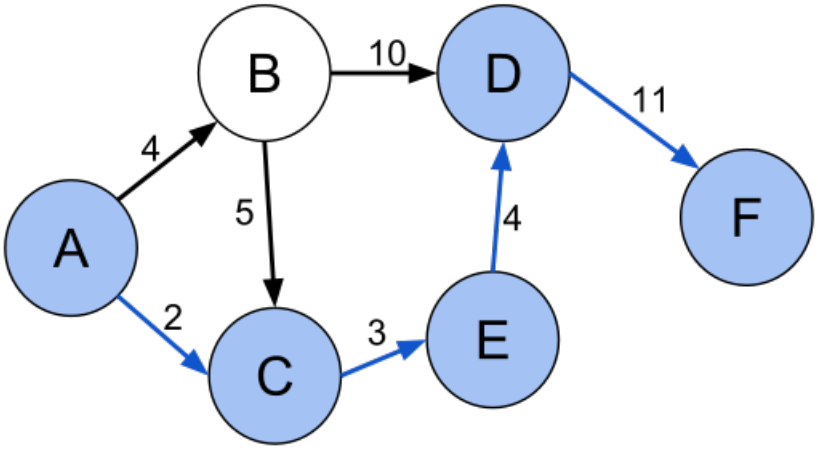
Releases

Current stable release is v1.0.1. See themilestones to check the current development status.

```
<dependencies>
 <!--
  Add this dependency under your pom.xml <dependencies> section to add
  all the dependencies of Hipster to your project. Add hipster-core
  instead of hipster-all for core functionality.
 -->
 <dependency>
  <groupId>es.usc.citius.hipster</groupId>
  <artifactId>hipster-all</artifactId>
  <version>1.0.1</version>
 </dependency>
</dependencies>
```

Quick Example

Let's solve the graph used inthis Wikipedia article about Shortest paths.



Although Hipster is graph agnostic, we include some useful classes to create a graph or a directed graph and the search problem. We create a graph using the GraphBuilder class and then we use the GraphSearchProblem to create the required components to solve it using Dijkstra's algorithm:

```
// Create a simple weighted directed graph with Hipster where
// vertices are Strings and edge values are just doubles
HipsterDirectedGraph<String,Double> graph =
    GraphBuilder.<String,Double>create()
     .connect("A").to("B").withEdge(4d)
     .connect("A").to("C").withEdge(2d)
     .connect("B").to("C").withEdge(5d)
     .connect("B").to("D").withEdge(10d)
     .connect("C").to("E").withEdge(3d)
     .connect("D").to("F").withEdge(11d)
     .connect("E").to("D").withEdge(4d)
     .createDirectedGraph();

// Create the search problem. For graph problems, just use
// the GraphSearchProblem util class to generate the problem with ease.
SearchProblem p = GraphSearchProblem
                      .startingFrom("A")
                      .in(graph)
                      .takeCostsFromEdges()
                      .build();

// Search the shortest path from "A" to "F"
System.out.println(Hipster.createDijkstra(p).search("F"));
```

Output result: Total solutions: 1 Total time: 6 ms Total number of iterations: 6 + Solution 1: - States: [A, C, E, D, F] - Actions: [2.0, 3.0, 4.0, 11.0] - Search information: WeightedNode{state=F, cost=20.0, estimation=0.0, score=20.0}

But that's not all. Hipster comes with different problem examples that illustrate how Hipster can be used to solve a wide variety of problems (not only graph search).

What's next?

If you want to learn how to solve a problem by searching with Hipster, check the wiki and the JavaDoc documentation. We also suggest you to check this presentation for a quick introduction.

License & Citation

This software is licensed under the Apache 2 license, quoted below.

Citation

This library was presented in the "9th Iberian Conference on Information Systems and Technologies (CISTI), 2014". If you use this library in your research projects, we encourage you to please cite our work:

> Rodriguez-Mier, P., Gonzalez-Sieira, A., Mucientes, M., Lama, M. & Bugarin, A. (2014).Hipster: An Open Source Java Library for Heuristic Search. *9th Iberian Conference on Information Systems and Technologies (CISTI)*.

```
@inproceedings{RodriguezMier2014,
 author = {Rodriguez-Mier, Pablo and Gonzalez-Sieira, Adrian and Mucientes, Manuel and and Lama, Manuel and Bugarin, Alberto},
 booktitle = {9th Iberian Conference on Information Systems and Technologies (CISTI 2014)},
 month = jun,
 volume = 1,
 title = {{Hipster: An Open Source Java Library for Heuristic Search}},
 pages = {481--486},
 isbn = "978-989-98434-2-4"
 address = "Barcelona",
 year = {2014}
}
```

INFORMACIÓN

Investigadores
Pablo Rodríguez Mier
Adrián González Sieira

Licenza

Como contribuír

DESCARGAR

- Repositorio Gitlab
- Descargar de Gitlab
- Repositorio Github

PUBLICACIÓNS

Hipster: An Open Source Java Library for Heuristic Search
9th Iberian Conference on Information Systems and Technologies (CISTI 2014), 2014