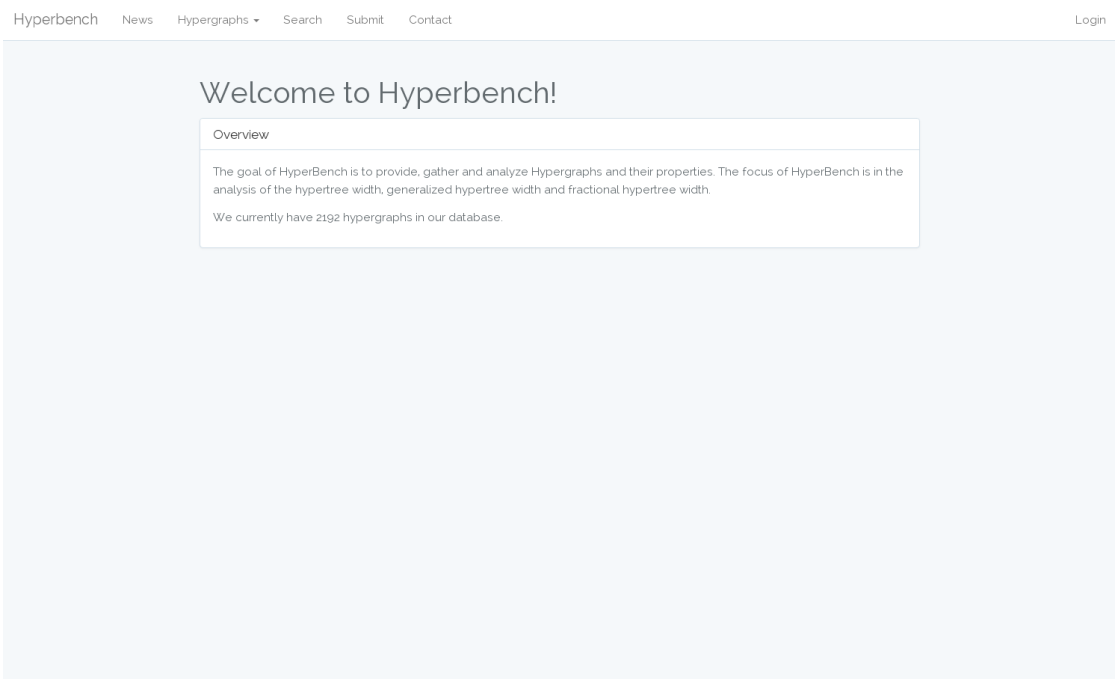
A Guide To HyperBench

W. Fischl

January 24, 2018



Contents

Changelog

Jan, 24th first preliminary version

Chapter 1

Welcome to HyperBench

HyperBench is a web platform to gather, analyze and publish hypergraphs, their corresponding hypertree width and further properties. In this short tutorial we explain the basic functionality that is available in the current version of HyperBench.

1.1 Menu Bar

The menu bar gives access to the following items:

- **News**
- **Hypergraphs** gives you access to all stored hypergraphs by category. The categories correspond to those given in [3]. Additionally, "All" lists all hypergraphs stored (Details in Section 1.2).
- **Search** (no functionality yet available)
- **Submit** (no functionality yet available)
- **Contact**

1.2 Hypergraph Category View

After selecting a hypergraph category, you are displayed the "Hypergraph Category View" (see Figure 1.1). In this view you see a list of all hypergraphs linked to this category. Each hypergraph can be linked to several categories (e.g. a "DaimlerChrysler" hypergraph is also a "CSP Other" hypergraph). The list of hypergraphs can be sorted by name, number of vertices and number of edges. With 

you are shown the "Hypergraph Details View" of the selected hypergraph (Details in Section 1.3). The "Download All" link allows you to download all hypergraphs in this category as a ZIP-file.

Summary of Properties. In order to view a summary of number of hypergraphs in this category having a specific property (*Degree*, *BIP*, *3-BIP*, *4-BIP* and *VC-dim*), click the "Summary of Properties" link. Click the link again to close the property page.

1.3 Hypergraph Details View

On this page you see the details of one hypergraph (see Figure 1.2). On the top you have the ability to show or download the current hypergraph. A hypergraph is displayed and stored as a text file with the following format:

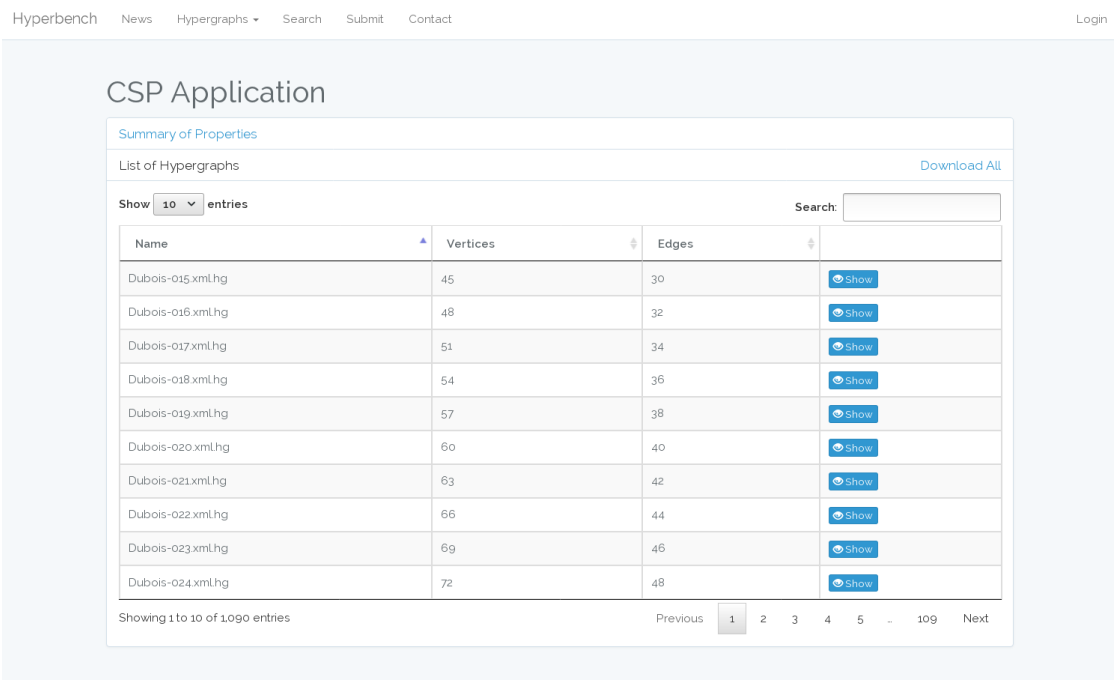


Figure 1.1 Hypergraph Category View

```
Hyperedge_1 (Vertex_1_1, Vertex_1_2, ..., Vertex_1_n1),

Hyperedge_2 (Vertex_2_1, Vertex_2_2, ..., Vertex_2_n2),

...

Hyperedge_1 (Vertex_m_1, Vertex_m_2, ..., Vertex_m_nm).
```

Due to this simple structure, we do not formally define the syntax of our format. Just note that the names of hyperedges and vertices may consist of any combination of lower- and uppercase letters, numbers, underscore, colon, etc. Comments start with '%' and continue until the end of the line. Some hypergraphs may also contain definitions within angle brackets in the header.

Properties. The properties tab shows the same properties summarized in the "Hypergraph Category View".

Hypertree Widths. A summary of the widths calculated by our various hypertree decomposition algorithms is displayed. The lower bounds for (generalized or fractional) hypertree width are calculated by looking for the highest width were a (generalized or fractional) hypertree decomposition has not been found. Having a lower bound for

fractional hypertree width implies the lower bound for generalized hypertree width and hypertree width and a lower bound for generalized hypertree width implies the lower bound for hypertree width. The upper bounds for (generalized) hypertree width are the smallest width were a(generalized or fractional) hypertree decomposition has been found. Having an upper bound for hypertree width implies the upper bound for generalized and fractional hypertree width and hypertree width and an upper bound for generalized hypertree width implies the upper bound for fractional hypertree width.

List of Hypertrees. The "List of Hypertrees" gives an overview of the tried algorithms on this hypergraph. The following columns are displayed:

- **ID**: an identifier for this hypertree
- **Hypertree**: If the algorithm returns *yes* then you can download the hypertree as a text file written in the Graph Modelling Language[1]. If the algorithm returns *no* or *times out*, "HT not found!" is displayed.
- **Width**: The lowest hypertree width the algorithm has found.
- **Time Used**: The time the algorithm ran in seconds.
- **Killed By Timeout**: Is true in case the algorithm didn't suceed to either find or not find a hypertree decomposition of the given width.
- **Date Created**: The date and time the algorithm was run.
- **Algorithm Name**: The name of the algorithm according to the algorithms explained in [3]:
  - newdetkdecomp: a back-tracking based algorithm to find hypertree decompositions (algorithm type: "h") [2]. The version 2.0 is an improved version allowing for extensions used by next algorithms.
  - globalbipkdecomp: an algorithm to find generalized hypertree decompositions (algorithm type: "g"). This algorithm runs in polynomial time if the hypergraph has the bounded intersection property (BIP). For details see [1, 3].
  - localbipkdecomp: same as globalbipkdecomp.
  - balsepkdecomp: an algorithm to find generalized hypertree decompositions (algorithm type: "g"). This algorithm was first described in [3].
  - improvehd: an algorithm to find fractional hypertree decompositions (algorithm type: "f") by improving existing hypertree decomposition using fractional edge covers. This algorithm was first described in [3].
  - bestimprovehd: see improvehd
  - fracimprovehd: see improvehd

[1] see `https://en.wikipedia.org/wiki/Graph_Modelling_Language` for a description. Use the yEd Graph Editor (`https://www.yworks.com/products/yed`) to display the hypertree

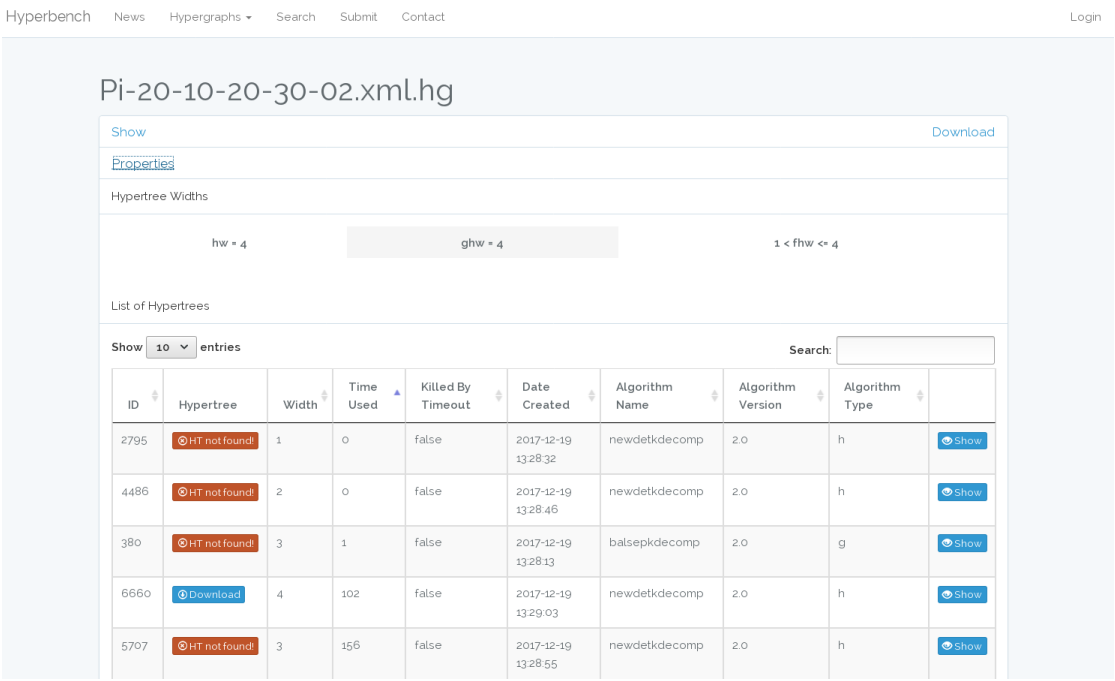


Figure 1.2 Hypergraph Details View

All algorithms are on GitHub: `https://github.com/TUfischl/newdetkdecomp`.

- **Algorithm Version**: a version number for the algorithm.
- **Algorithm Type**: Either “h”, “g” or “f” depending on whether the algorithm outputs a hypertree decomposition, a generalized hypertree decomposition or a fractional hypertree decomposition of the hypergraph.
- Show: Shows the “Hypertree View”, see Section 1.4.

1.4 Hypertree View

The Hypertree View displays the hypertree for a specific hypergraph (see Figure 1.3). As before you are able to download the hypertree as a text file written in the Graph Modelling Language. Additionally, all properties given above (in Section 1.3) can also be viewed here.

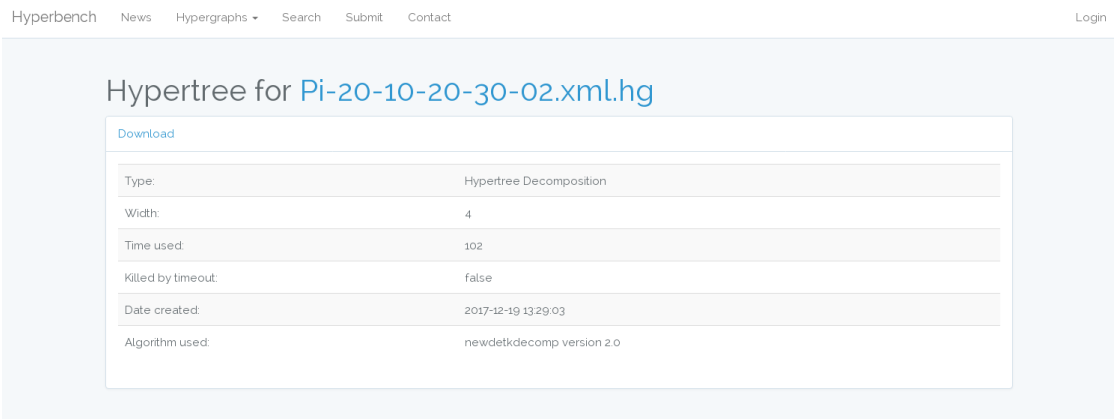


Figure 1.3 Hypergraph Details View

Bibliography

[1] Wolfgang Fischl, Georg Gottlob, and Reinhard Pichler. General and fractional hypertree decompositions: Hard and easy cases. In *Proc. PODS 2018 (to appear)*, 2018.

[2] Georg Gottlob and Marko Samer. A backtracking-based algorithm for hypertree decomposition. *ACM Journal of Experimental Algorithmics*, 13, 2008.

[3] Davide M. Longo Wolfgang Fischl, Georg Gottlob and Reinhard Pichler. HyperBench: A benchmark and tool for hypergraphs and empirical findings. forthcoming report, 2018.