




# Interactive Bin Packing: A Java Application for Learning Constructive Heuristics for Combinatorial Optimization

Vincent A. Cicirello<sup>1</sup>

<sup>1</sup> Computer Science, School of Business, Stockton University, Galloway, NJ 08205

DOI: [10.21105/jose.00140](https://doi.org/10.21105/jose.00140)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

**Submitted:** 03 July 2020

**Published:** 31 March 2022

## License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](https://creativecommons.org/licenses/by/4.0/)).

## Summary

Interactive Bin Packing provides a self-guided tutorial on combinatorial optimization, the bin packing problem, and constructive heuristics. It also enables users to interact with bin packing instances to explore their own problem solving strategies, or to test their knowledge of the constructive heuristics covered by the tutorial. The application is not a solver for bin packing, but rather it is a tool for learning about the bin packing problem, and for learning about heuristic techniques for solving instances of the problem.

The repository (<https://github.com/cicirello/InteractiveBinPacking>) contains the source code, documentation, and example assignments suitable for courses on discrete mathematics, algorithms, or artificial intelligence (AI), or by self-directed learners. An executable jar of the application is available from GitHub Releases and Maven Central.

## Bin Packing

Bin packing is a combinatorial optimization problem involving packing a set of items into a set of bins to minimize the number of bins. It has many practical applications, including within logistics, such as packing shipments into trucks or airplanes, as well as applications such as creating compilations of musical works, scheduling television advertisements, placing files in fixed-length memory blocks on a file system, wireless network channel scheduling, job scheduling, among many others ([Leinberger et al., 1999](#); [Malkevitch, 2004a, 2004b](#); [Wang et al., 2006](#)).

As an NP-Hard ([Garey & Johnson, 1979](#)) optimization problem, it is unlikely that any polynomial-time algorithm exists guaranteed to optimally solve bin packing. Thus, heuristics and metaheuristics are often used. Among the topics covered by this tutorial application are constructive heuristics. A heuristic is a practical problem solving approach that provides a solution of sufficient quality, while using relatively little time. A constructive heuristic iteratively applies some rule to build upon an initially empty solution until it is complete.

## Origin of the Application

In my teaching, I have used bin packing to introduce heuristics to students in courses for computer science majors, as well as for non-majors. Using the Computer Science Unplugged ([Nishida et al., 2009](#)) methodology, I previously developed an in-class activity called Collective Bin Packing ([Cicirello, 2009](#)), in which students take turns making

individual decisions on which item to place in which bin. One objective of that activity is to illustrate concepts of collective intelligence, or swarm intelligence, where intelligent behavior emerges from indirect interaction within the swarm. A preliminary version of Interactive Bin Packing was used by the instructor to keep track of the state of the emerging solution. Without the aid of the application, the Collective Bin Packing activity was later adapted to an online course environment (Cicirello, 2013), and even used in a course for non-majors (Cicirello, 2008).

## Functionality

This first public release of Interactive Bin Packing features a self-guided tutorial that explains key concepts of combinatorial optimization, bin packing, and constructive heuristics. It also includes several heuristic modes, in which the user can test their knowledge of the different constructive heuristics. When used in a heuristic mode, the user is expected to make decisions that match their chosen heuristic's behavior. The application provides feedback on correctness throughout.

## Target Audience, Instruction Time, and Objectives

The application is a software tool to assist students in learning. It assumes prerequisite knowledge of introductory discrete mathematics concepts including set theory and functions. It can be used in courses on: (a) discrete mathematics toward the end to introduce combinatorial optimization, (b) algorithms to provide an example of an NP-Hard problem and the use of heuristics to efficiently compute satisficing solutions despite the problem's complexity, or (c) AI as an example of heuristic problem solving or prior to coverage of local search algorithms. It can also be used by self-directed learners.

The GitHub repository also contains a directory with example assignments that utilize the application. One of these is designed to provide an introduction to combinatorial optimization and heuristic problem solving, and requires no pre-lecture. The time to complete this assignment, including working through the self-guided tutorial within the application and completing self-check exercises, varies between 66 minutes and 135 minutes, depending upon the learner's prior background and course level. To use the application in your own teaching, we recommend utilizing this assignment as an introduction to your topic of choice. For example, in an AI course, you might then cover hill climbing to show how you can iteratively improve upon the heuristic's solution; or in an algorithms course, you might use this assignment while covering the theory of NP-Completeness.

The objectives of the Interactive Bin Packing Application include:

- gaining a general understanding of combinatorial optimization;
- gaining an understanding of the bin packing problem, its applications, and how it is an example of combinatorial optimization;
- learning about lower bounds;
- learning about constructive heuristics;
- learning about the most common constructive heuristics for bin packing, including first-fit, best-fit, first-fit decreasing, and best-fit decreasing; and
- serving as an interactive environment for students (whether in a formal course context, or for informal self-directed learning) to explore problem solving methods for combinatorial optimization.

## Statement of Need

Many combinatorial optimization problems, including bin packing, traveling salesperson, largest common subgraph, and numerous others, are NP-Hard; and at the same time have practical real-world applications requiring quickly producing solutions. Heuristics, meta-heuristics, and other approximate algorithms enable balancing the theoretical hardness of NP-Hard problems with the need for quality solutions to those problems. Constructive heuristics provide a gentle introduction to this category of problem solving techniques. Although there are many tutorials about, or that use, bin packing, the existing tutorials are limited to web applications that calculate heuristic solutions (e.g., [PlanetCalc, 2021](#)), algorithm animations that visualize heuristics in action (e.g., [Černohorská, 2015](#)), and optimization library tutorials that use bin packing to provide examples of using an optimization API (e.g., [Google Developers, 2021](#)). However, existing bin packing tutorials lack interactivity as well as feedback to the learner. Immediate feedback in interactive environments is an effective pedagogical tool for computer science, such as in courses on introductory programming ([Kim & Ko, 2017](#); [Price et al., 2017](#)), data structures ([Dicheva et al., 2019](#)), algorithms ([Deb et al., 2017](#)), software engineering ([Krusche & Seitz, 2018](#)), among others. Puzzle-like problems have been shown effective in engaging students in algorithmic thinking ([Levitin, 2005](#)). The interactive environment of our Interactive Bin Packing application keeps learners engaged with such a puzzle-like experience, while utilizing pedagogical best practices such as instant automated feedback.

## References

- Cicirello, V. A. (2008). An interdisciplinary course on artificial intelligence designed for a liberal arts curriculum. *Journal of Computing Sciences in Colleges*, 23(3), 120–127. <https://dl.acm.org/doi/abs/10.5555/1295109.1295137>
- Cicirello, V. A. (2009). Collective bin packing: An active learning exercise. *Journal of Computing Sciences in Colleges*, 24(6), 117–123. <https://dl.acm.org/doi/abs/10.5555/1529995.1530020>
- Cicirello, V. A. (2013). A CS unplugged activity for the online classroom. *Journal of Computing Sciences in Colleges*, 28(6), 162–168. <https://dl.acm.org/doi/abs/10.5555/2460156.2460187>
- Černohorská, V. (2015). *Heuristics for 1D and 2D bin packing*. <http://www.binpacking.4fan.cz/>
- Deb, D., Fuad, M. M., & Kanan, M. (2017). Creating engaging exercises with mobile response system (MRS). *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, 147–152. <https://doi.org/10.1145/3017680.3017793>
- Dicheva, D., Irwin, K., & Dichev, C. (2019). OneUp: Engaging students in a gamified data structures course. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 386–392. <https://doi.org/10.1145/3287324.3287480>
- Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of NP-completeness*. W. H. Freeman. ISBN: 0716710455
- Google Developers. (2021). *OR-tools: The bin packing problem*. [https://developers.google.com/optimization/bin/bin\\_packing](https://developers.google.com/optimization/bin/bin_packing)
- Kim, A. S., & Ko, A. J. (2017). A pedagogical analysis of online coding tutorials. *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, 321–326. <https://doi.org/10.1145/3017680.3017728>

- Krusche, S., & Seitz, A. (2018). ArTEMiS: An automatic assessment management system for interactive learning. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 284–289. <https://doi.org/10.1145/3159450.3159602>
- Leinberger, W., Karypis, G., & Kumar, V. (1999). Multi-capacity bin packing algorithms with applications to job scheduling under multiple constraints. *Proceedings of the 1999 International Conference on Parallel Processing*, 404–412. <https://doi.org/10.1109/ICPP.1999.797428>
- Levitin, A. (2005). Analyze that: Puzzles and analysis of algorithms. *Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education*, 171–175. <https://doi.org/10.1145/1047344.1047409>
- Malkevitch, J. (2004a). Bin packing. *Feature Column from the AMS: Monthly Essays on Mathematical Topics*. <http://www.ams.org/publicoutreach/feature-column/fcarc-bins1>
- Malkevitch, J. (2004b). Bin packing and machine scheduling. *Feature Column from the AMS: Monthly Essays on Mathematical Topics*. <http://www.ams.org/publicoutreach/feature-column/fcarc-packings1>
- Nishida, T., Kanemune, S., Idosaka, Y., Namiki, M., Bell, T., & Kuno, Y. (2009). A CS unplugged design pattern. *Proceedings of the 40th ACM Technical Symposium on Computer Science Education*, 231–235. <https://doi.org/10.1145/1508865.1508951>
- PlanetCalc. (2021). *Online calculator: Bin packing problem*. <https://planetcalc.com/917/>
- Price, T. W., Dong, Y., & Lipovac, D. (2017). ISnap: Towards intelligent tutoring in novice programming environments. *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, 483–488. <https://doi.org/10.1145/3017680.3017762>
- Wang, C., Li, B., Sivalingam, K., & Sohraby, K. (2006). Scalable multiple channel scheduling with optimal utility in wireless local area networks. *Wireless Networks*, 12(2), 189–198. <https://doi.org/10.1007/s11276-005-5266-y>