The Journal of Open Source Software

kep_solver: A Python package for kidney exchange programme exploration

William Pettersson [1]

1 School of Computing Science, University of Glasgow, UK

DOI: 10.21105/joss.04881

Software

- Review
- Repository
- Archive

Editor: Jacob Schreiber

Reviewers:

- @arianesasso
- @igarizio

Submitted: 22 September 2022
Published: 27 November 2022

Summary

Kidney disease is one of the top ten leading causes of death globally (*WHO Global Health Estimates*, 2019), and unfortunately has no known cure. Instead, late-stage kidney disease is treated with either dialysis or a donated kidney transplant. Of these, a kidney transplant is cheaper, and offers both a better quality of life and a longer life expectancy (Axelrod et al., 2018). Such donor kidneys can come from either living or deceased donors, with living donor transplants resulting in better outcomes for the recipient (Hart et al., 2017; Wolfe et al., 2010). However finding a living donor who is both willing to donate and medically compatible can be difficult.

Kidney exchange programmes (KEPs) greatly increase the rate of living donor kidney transplants by alleviating the requirement that a willing donor must be medically compatible with their chosen recipient. Instead, recipients still pair with at least one willing donor, but transplants are organised such a donor donates a kidney to a recipient if and only if their paired recipient receives a kidney. In particular, a donor will often not donate to their paired recipient.

The question that arises is then: given a number of recipients, along with their paired donors, which transplants should be selected to go ahead? This is one of the problems that KEPs solve. Commonly, this is solved by first building a compatibility graph: a graph that represents all the donor-and-recipient pairs as well as arcs indicating that there is a potential for a transplant from a donor to a recipient. Then a set of vertex-disjoint cycles are selected through an integer programme according to some pre-determined criteria (e.g. maximising number of transplants, maximising transplants to hard-to-match recipients). These selected cycles correspond to sets of transplants that have been selected to proceed, and will undergo further checks for medical compatibility and transplant procedures.

Statement of need

Almost all research-based software for kidney exchange that has been published has been written from scratch for one particular paper but without re-use in mind (including projects from this author). See for instance:

- Delorme, García, Gondzio, Kalcsics, Manlove, & Pettersson (2022) Code
- Delorme, García, Gondzio, Kalcsics, Manlove, Pettersson, & Trimble (2022) Code
- Toulis & Parkes (2015) Code
- Dickerson et al. (2014) Code

Such papers, however, form the minority as many papers in the field don't even publish their code. This has unnecessarily increased the workload for researchers, and also increases potential for inaccurate results due to coding errors. `kep_solver` reduces or completely removes this burden from researchers by providing a framework that can be built upon. This framework is useful for a wide variety of research goals, including:

- Improved algorithms and models for finding solutions,
- Improved outcomes by adapting or changing the goals, targets, and parameters of the KEP, and
- Improved simulation of programmes by better statistical analyses of real-world kidney exchange programme populations.

By providing a stable and standard framework that is easy to use and build upon, `kep_solver` will also accumulate the latest innovations in kidney exchange programmes, ensuring not only that researchers can easily implement their own improvements, but also compare said improvements to the state-of-the-art in the field.

Functionality

`kep_solver` includes code for all components required to form the basis of a real-world kidney exchange programme. It includes code that handles file input/output for the currently-public formats, which are also documented in `kep_solvers` own documentation. It manages donors, recipients, and transplants, including ensuring that properties such as blood groups and cPRA are valid upon entry. It can construct and analyse the compatibility graphs that map the potential transplants between donors and recipients, including enumeration of all potential cycles and chains. It contains a number of different optimisation criteria that can be configured in arbitrary hierarchies. It includes a cycle-and-chain integer programming model for finding an optimal set of transplants according to a configured hierarchy of criteria, and solves said model via PuLP. It can also create a number of different random entity generators, ranging from blood group generators to complete instance generators. Each such generator can be configured with an appropriate distribution based on examination of real-world instances.

Sample usage

The simplest usage of `kep_solver` is to run a single instance through a simple kidney exchange programme, so we demonstrate that now.

```
from kep_solver.fileio import read_json
from kep_solver.pool import Pool
from kep_solver.model import TransplantCount

instance = read_json("input.json")
pool = Pool([TransplantCount()],
            description="My first KEP Pool",
            maxCycleLength=3,
            maxChainLength=2)
solution, model = pool.solve_single(instance)
num_transplants = sum(len(modelled.exchange) for modelled in solution.selected)
print(f"A total of {len(solution.selected)} exchanges were found")
print(f"These represent {num_transplants} transplants")
```

The first three lines simply import the necessary portions of `kep_solver`. We then read an instance from a file, create a pool with one optimisation objective (`TransplantCount()`), solve the single given instance and print out how many transplants are selected.

Acknowledgements

This software has been supported by the Engineering and Physical Sciences Research Council (EPSRC) grant EP/T004878/1 (Multilayer Algorithmics to Leverage Graph Structure).

References

Axelrod, D. A., Schnitzler, M. A., Xiao, H., Irish, W., Tuttle-Newhall, E., Chang, S.-H., Kasiske, B. L., Alhamad, T., & Lentine, K. L. (2018). An economic asssessment of contemporary kidney transplant practice. *Americal Journal of Transplantation*, *18*, 1168–1176. https://doi.org/10.1111/ajt.14702

Delorme, M., García, S., Gondzio, J., Kalcsics, J., Manlove, D., & Pettersson, W. (2022). New algorithms for hierarchical optimisation in kidney exchange programmes. *Operations Research (Accepted, to Appear)*.

Delorme, M., García, S., Gondzio, J., Kalcsics, J., Manlove, D., Pettersson, W., & Trimble, J. (2022). Improved instance generation for kidney exchange programmes. *Computers & Operations Research*, *141*, 105707. https://doi.org/10.1016/j.cor.2022.105707

Dickerson, J. P., Procaccia, A. D., & Sandholm, T. (2014). Price of fairness in kidney exchange. *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems*, 1013–1020. https://doi.org/10.5555/2615731.2617407

Hart, A., Smith, J. M., Skeans, M. A., Gustafson, S. K., Stewart, D. E., Cherikh, W. S., Wainright, J. L., Kucheryavaya, A., Woodbury, M., Snyder, J. J., Kasiske, B. L., & Israni, A. K. (2017). OPTN/SRTR 2015 annual data report: kidney. *American Journal of Transplantation*, *17*, 21–116. https://doi.org/10.1111/ajt.14124

Toulis, P., & Parkes, D. C. (2015). Design and analysis of multi-hospital kidney exchange mechanisms using random graphs. *Games and Economic Behavior*, *91*, 360–382. https://doi.org/j.geb.2015.01.001

WHO Global Health Estimates. (2019). https://www.who.int/data/global-health-estimates

Wolfe, R. A., Roys, E. C., & Merion, R. M. (2010). Trends in organ donation and transplantation in the United States, 1999–2008. *American Journal of Transplantation*, *10*, 961–972. https://doi.org/10.1111/j.1600-6143.2010.03021.x