Paper about inference and ops-search. Created by Simon and Saul so it’s a highly relevant and accurate paper: <https://www.cognesence.co.uk/downloads/Clojure_Tools_for_Practical_Artificial_Intelligence.pdf>

Symbolic pattern matching in clojure. Really useful for understanding what can be done with matcher methods (e.g. ?x, ??x, ?\_, mlet, mout, mcond...etc). Created by Simon:

<https://research.tees.ac.uk/ws/files/4269909/608485.pdf>

Details “SIR”, “STUDENT”, and “SHRDLU” which have been mentioned in lectures.

More pattern matching in clojure. Similar to the above paper but shows additional features and more complicated examples. Created by Simon:

<https://www.scm.tees.ac.uk/isg/website/pubs/ELS-2017-Lynch(1).pdf>

Breadth-first search:

<https://www.sciencedirect.com/science/article/pii/S0004370205002158>

Breadth-first search:

<https://www.sciencedirect.com/science/article/pii/S0743731513001135>

Breadth-first search:

<https://doaj.org/article/2ae3f6a0f75f4eceac1d03248ef0a8f4>

Breadth-first search

<https://www.hindawi.com/journals/sp/2013/702694/abs/>

Efficiency of Algorithms -

<https://www.sciencedirect.com/science/article/pii/S0360131503000848>

Performance Evaluation of Inference Engine in Static and Changeable Environment

<http://www.ijcsit.com/docs/Volume%203/vol3Issue3/ijcsit2012030375.pdf>

Designing a knowledge-based system for benchmarking

<https://www.sciencedirect.com/science/article/pii/S0950705111000335>

Rapid Benchmarking for Semantic Web Knowledge Base Systems

<https://drive.google.com/open?id=1U9YjuLdmCBkueV0Tkns-FObhyWtbyqT8>

Efficient benchmark

<https://www.sciencedirect.com/science/article/pii/S0140366496010948>

Comparative Study of Complexities of Breadth- First Search and Depth-First Search Algorithms using Software Complexity Measures

<http://www.iaeng.org/publication/WCE2010/WCE2010_pp203-208.pdf>

Uninformed Search - There is a book in the library that talks about these metrics as well

<https://www.cs.colostate.edu/~howe/cs440/csroo/yr2015fa/more_progress/04_uninformed_search2015.pdf>

(specifically the Metrics for comparing search strategies slide)

TODO

Breadth vs Depth (look for strengths/weaknesses).

Inference engines?

-- NOT REFERENCE PAPERS, MAY BE USEFUL --

-- Dijkstra Algo, If needed --

<https://www.programiz.com/dsa/dijkstra-algorithm>

-- Clojure, time --

<https://clojuredocs.org/clojure.core/time>

-- Criterium, clojure, alternative to time f --

<http://clojure-goes-fast.com/blog/benchmarking-tool-criterium/>

-- Clojure memory usage --

<https://stackoverflow.com/questions/26213464/what-is-the-best-way-to-measure-how-much-memory-a-clojure-program-uses>

<https://github.com/clojure-goes-fast/clj-memory-meter>