NetworkX Backend Dispatching

- Plugin Arch.
- entry_points
- Automatic backend testing

For testing:

```
NETWORKX_TEST_BACKEND=parallel
NETWORKX_FALLBACK_TO_NX=True
pytest --pyargs networkx "$@"
```

9th July, 2024

config

can_run
should_run
on_start_tests
caching
logging
backend priority

NetworkX

A pure Python library for network analysis

backends

Backend Interface

backend_info

get_info

nx-parallel

A parallel backend for NetworkX, utilizing Joblib to implement graph algorithms on multiple CPU cores.

too many parallel processes and no GIL

To chunk or not to chunk?

Usage:

```
>>> import nx_parallel as nxp
>>> nx.betweenness_centrality(G,
backend="parallel")
>>> H = nxp.ParallelGraph(G)
>>> nx.betweenness_centrality(H)

$ export NETWORKX_AUTOMATIC_BACKEND=parallel
  && python nx_code.py
```

"Speed-ups"

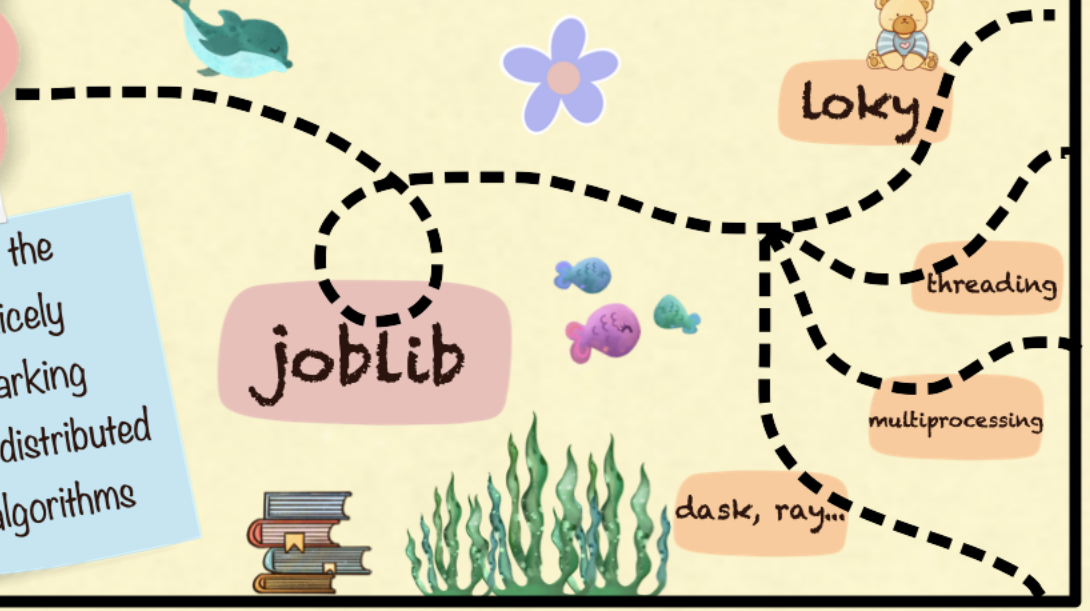
local_efficiency-**8.8x**
betweenness_centrality-**6.4x**
square_clustering-**4.1x**
bipartite.node_redundancy-**4.1x**

* for 300-node random graphs (edge probability = 0.6)

feel free to contribute and nurture!

TO-DO

- Designing the pipeline nicely
- Benchmarking
- Adding distributed graph algorithms



Aditi Juneja (@Schefflera-Arboricola)