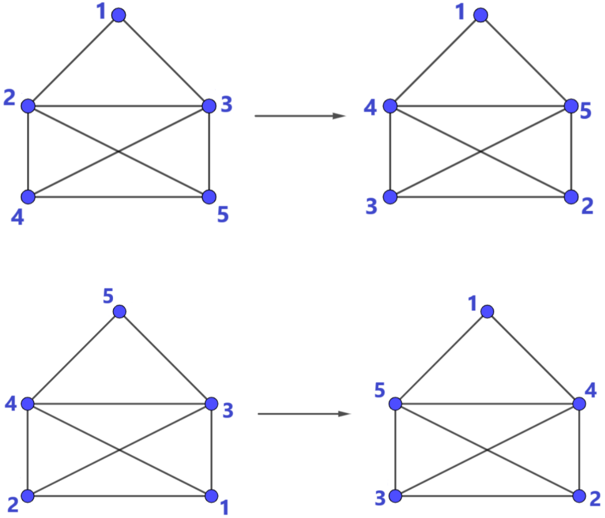
Graph Canonization Algorithms

Nagy Szabolcs

2024/25 1st semester, Math Project 3

Canonization overview

Isomorphism-invariant graph labeling.



Motivation and goal of project

Applications:

- **Efficient generation of non-isomorphic graphs ($n < 20$)**,
- deciding isomorphism between graphs,
- various uses in biology in and chemistry,
- etc. . .

Goal: examine and attempt to improve upon known canonization algorithms.

McKay's algorithm

A search-tree of *ordered partitions* on $V(G)$

- Root contains unit partition: $\Pi_0 = V(G)$.
- Partition of a child is always finer than that of the parent: $\Pi_t \subseteq \Pi_{\text{parent}(t)}$.
- Leaves are trivial partitions $\iff$ permutations: $\Pi_l = \{v_{\pi_1}\}, \{v_{\pi_2}\}, \ldots \{v_{\pi_n}\}$ for some $\pi \in S_n$.

Graph degree information and automorphism data are used to decrease the size of the search-tree.

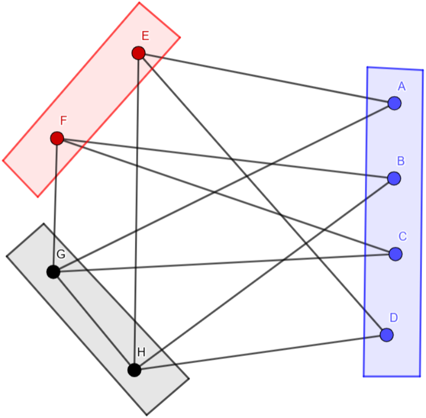
Equitability

A partition Π is equitable if:
$\forall X_1, X_2 \in \Pi, \forall v_1, v_2 \in X_1 : d(v_1, X_2) = d(v_2, X_2)$

When setting Π_t, if it is not equitable, *refine* it so it is.

Possible to do in isomorphism-invariant manner.

Equitable partition example

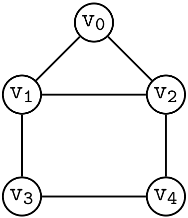


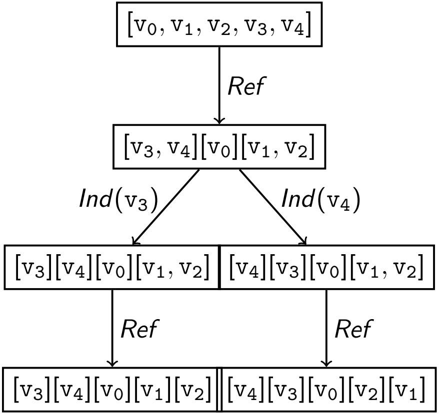
$$V_1 = \{A, B, C, D\},$$
$$V_2 = \{E, F\},$$
$$V_3 = \{G, H\}$$

Degrees:

$$\begin{bmatrix} 0 & 1 & 1 \\ 2 & 0 & 1 \\ 2 & 1 & 1 \end{bmatrix}$$

Refinement in action





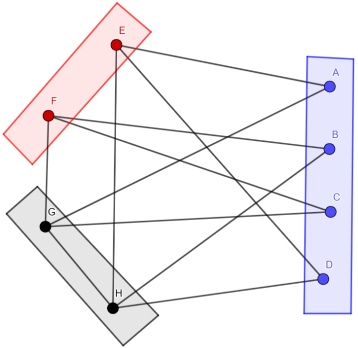
k-equitability

- $d_k(v, X) = |\{w \in X : \exists v \to w \text{ walk } Q : |Q| = k\}|$

- A partition Π is k-equitable if for any $\ell = 1 \ldots k$:
  $\forall X_1, X_2 \in \Pi, \forall v_1, v_2 \in X_1 : d_\ell(v_1, X_2) = d_\ell(v_2, X_2)$

- Notice: 1-equitability $\iff$ equitability

Equitable, but not 2-equitable partition



$V_1 = \{A, B, C, D\}$,
$V_2 = \{E, F\}$,
$V_3 = \{G, H\}$

$d_2(A, V_3) = d_2(B, V_3) = 1$
$d_2(C, V_3) = d_2(D, V_3) = 2$

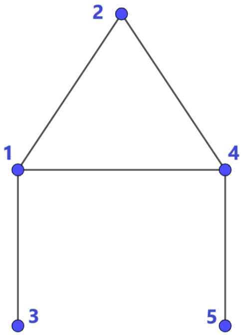
Markov-chains, stationary distributions

Construct Markov-chain from graph:

- $I = V(G)$,
- step into any neighbor with equal probability.

Idea: calculate a stationary distribution for the above Markov-chain, get ordered partition by sorting elements based on distribution value.
This involves finding the solution of the lin. eq. sys.
$\begin{bmatrix} P^T - I \\ \mathbb{1} \end{bmatrix} \mu = \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix}$, where P is the transition matrix.

Basic Markov-chain example



Transition matrix:

$$\begin{bmatrix} 0 & 0.33 & 0.33 & 0.33 & 0 \\ 0.5 & 0 & 0 & 0.5 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0.33 & 0.33 & 0 & 0 & 0.33 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Unique stationary distribution:
$\mu = \frac{1}{10}\{3, 2, 1, 3, 1\}$

Resulting ordered partition:
$\Pi = \{3, 5\}, \{2\}, \{1, 4\}$

Ensuring isomorphism-invariance

Problem: stationary distribution might not be unique.
Enough to ensure uniqueness: irreducibility, aperiodicity.
Solution: additional "connecting" vertex z:

- $V(G') = V(G) + z$
- $zv \in E(G')$ for all $v \in V(G')$

Constructing the same Markov-chain for G' always yields a unique stationary distribution.

Improving the Markov-chain

More information can be gleamed from the distribution by putting more thought into the construction:

- adjust probabilities based on the current partition, relations between neighboring cells
  (allows repeated use of distributions),
- possible steps based on k-long paths between vertices
  (helps in “ensuring” k-equitability).

Results

Advantages of using k-equitability and stationary distributions:

- Can find finer partitions than regular equitability checking
- Decreases the number of search-tree nodes for certain graphs, reducing runtime.

Drawbacks:

- More time spent examining each node
- Search-tree not guaranteed to be reduced, many graphs are unaffected
- Does not notably improve upon automorphism pruning

Canonizing all labeled graphs of size 6

37268 graphs total

- 2E - 2-equitability refinement
- SD - Stationary distribution
- AP - automorphism pruning

Method	Nodes	Runtime	# smaller trees	# bigger trees
$\emptyset$	139 995	770 ms	-	-
2E	139 935	740 ms	60	0
SD	139 770	754 ms	195	0
AP	127 915	684 ms	-	-
AP + 2E	127 915	730 ms	0	0
AP + SD	128 215	720 ms	0	90

Our work in this semester

What we did:

- maintain/optimize our canonization implementation, and the corresponding graph generator (C++)
- implement cell refinement based on 2-equitability,
- implement cell refinement based on stationary distributions.

Plans for the future:

- generalize cell refinement for k-equitability,
- further optimize Markov-chain construction,
- look into further potential improvements to canonization.

Closing thoughts

In the last 3 semesters, we have:

- thoroughly studied the theory of canonical labelings,
- comprehended the main algorithm and its implementation,
- produced our own object-oriented canonization program,
- used it to create our own customizable isomorphism-free graph generator,
- tested some potential improvements to the algorithm,
- found solid ground as to where to improve in the future.