Complex-Network Modelling and Inference

Lecture 17: Operations on graphs (binary operators)

Matthew Roughan
<matthew.roughan@adelaide.edu.au>
https://roughan.info/notes/Network_Modelling/

School of Mathematical Sciences,
University of Adelaide

March 7, 2024

Section 1

Binary operators

Binary Operators

- Disjoint union $G \cup H$
- Graph *products* based on the Cartesian product of the vertex sets:
  - Cartesian product $G \square H$
  - Tensor product $G \times H$
  - Strong product $G * H$
  - Lexicographic product $G \bullet H$
  - Rooted product $G \circ H$
- Others (not discussed here)
  - Clique sum
  - Corona and Zig-zag products
  - Series and Parallel compositions

Disjoint union $G \cup H$

- For two graphs G and H with disjoint node sets, *i.e.*,

  $$N(G) \cap N(H) = \phi$$

  the **disjoint union** $G \cup H$ is the graph formed by taking the union of the nodes and edges, *i.e.*,

  $$\begin{aligned} N(G \cup H) &= N(G) \cup N(H) \\ E(G \cup H) &= E(G) \cup E(H) \end{aligned}$$

- Properties
  - Commutative (for unlabelled graphs)
  - Associative (for unlabelled graphs)
- **Graph join:** disjoint union with all edges that join nodes from G to H

Cartesian product of vertices/nodes

- Cartesian (or direct) product defined on two sets X and Y
- Cartesian product of two sets of nodes results in all pairs of nodes with one from each set

$$X \times Y = \{(x, y) \mid x \in X \text{ and } y \in Y\}$$

  - its just a generalised vector
- Number of members of product

$$|X \times Y| = |X| \times |Y|$$

- Generalizes to n-ary products

Properties of Cartesian Products

- Associative (effectively)

$$X \times (Y \times Z) = (X \times Y) \times Z$$

- Doesn't commute $X \times Y \neq Y \times X$
  - order is important
  - in some of what follows we can ignore order because unlabelled graphs are isomorphic
- Distributive over intersections

$$A \times (B \cap C) = (A \times B) \cap (A \times C)$$

and unions

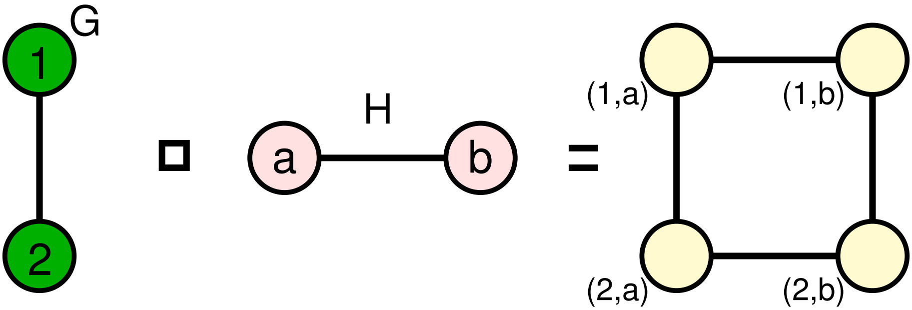
$$A \times (B \cup C) = (A \times B) \cup (A \times C)$$

Cartesian product of graphs

- $N(G \Box H) = N(G) \times N(H)$
- any two vertices $(u, u') \in G \Box H$ and $(v, v') \in G \Box H$ are adjacent iff one of the following is true
  - $u = v$ and $(u', v') \in E(H)$; or
  - $u' = v'$ and $(u, v) \in E(G)$

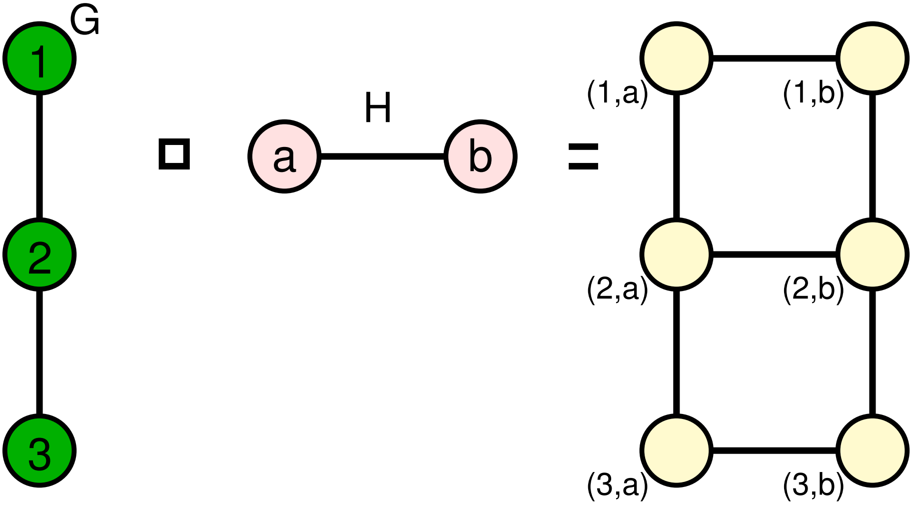
Example Cartesian Product 1

The Cartesian product of two (single) edges is a cycle with four vertices



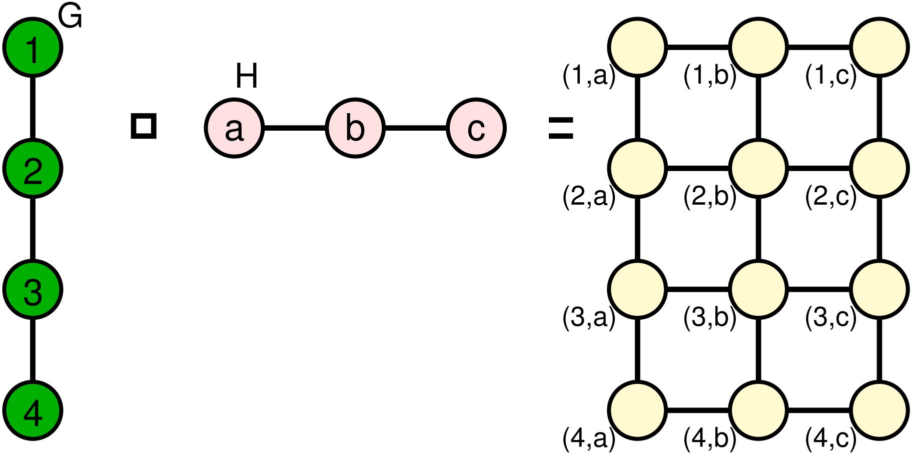
Example Cartesian Product 2

The Cartesian product of an single edge and a path graph is a ladder graph



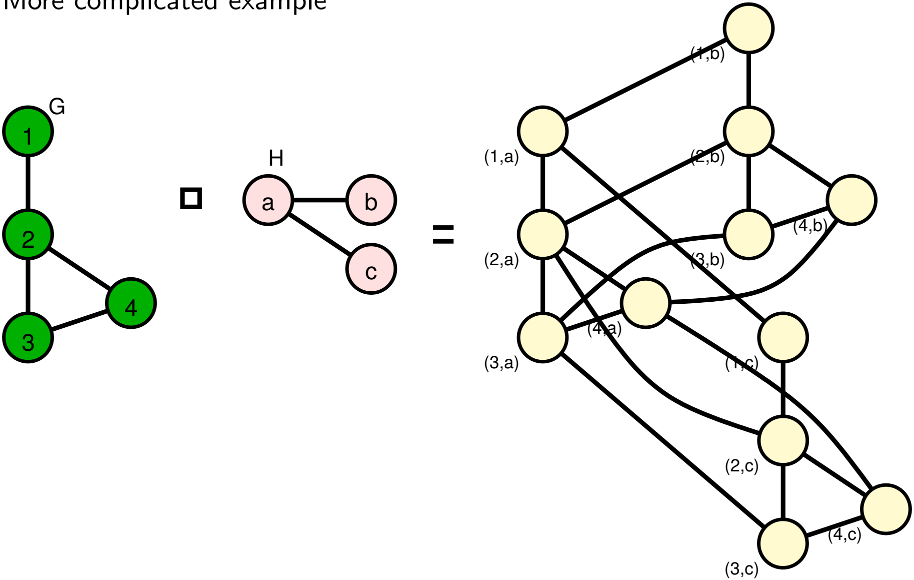
Example Cartesian Product 3

The Cartesian product of two path graphs is a grid graph.



Example Cartesian Product 4

More complicated example



Properties Cartesian product of graphs

- Commutes in the sense that $G \square H \simeq H \square G$
- Associative in the sense that $F \square (G \square H) \simeq (F \square G) \square H$
- Square symbol $\square$ used because Cartesian product of two edges is a "box" (a cycle with four edges).
- A Cartesian product is bipartite if and only if each of its factors is.

Cartesian product: Why?

- Ladder graphs approximate connectivity in some networks

- bi-connectivity is easy to achieve in a simple "cookie cutter" manner

Kronecker or Tensor product $A \otimes B$

Kronecker product of matrices A and B

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix}$$

- bi-linear and associative
- non-commutative

$$A \otimes B \neq B \otimes A \text{ (in general)}$$

- transposition is distributive over Kronecker product

$$(A \otimes B)^T = A^T \otimes B^T$$

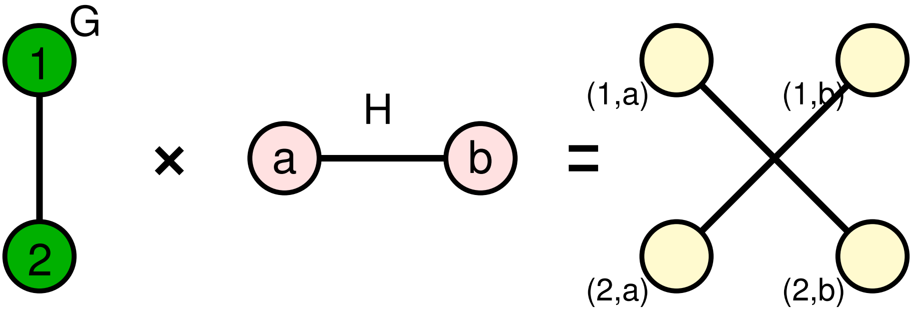
- lots of other well-known properties
  See http://en.wikipedia.org/wiki/Kronecker_product

Tensor product of graphs $G \times H$

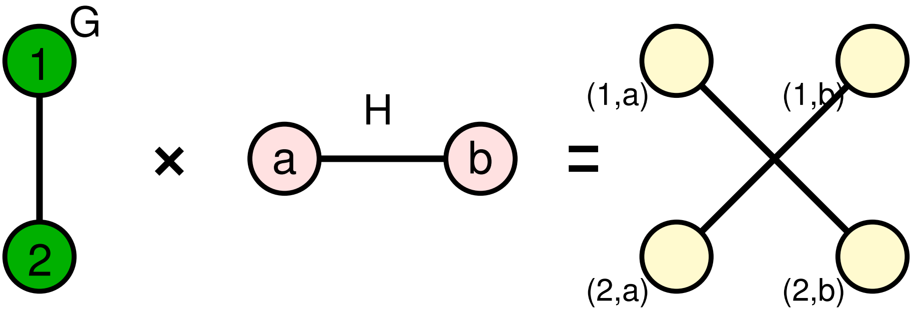
- Tensor product (direct product, categorical product, cardinal product, or Kronecker product) $G \times H$
- Defined by
  - $N(G \times H) = N(G) \times N(H)$
  - any two vertices (u, u') and (v, v') are adjacent iff $(u', v') \in E(H)$ and $(u, v) \in E(G)$
  - That is u' is adjacent to u in G and v' is adjacent to v in H.
- Equivalent to taking the Kronecker (or tensor) product of the adjacency matrices of G and H.

$$A_{G \times H} = A_H \otimes A_G$$

Example Tensor product



Tensor product by adjacency matrices



$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

Tensor product properties

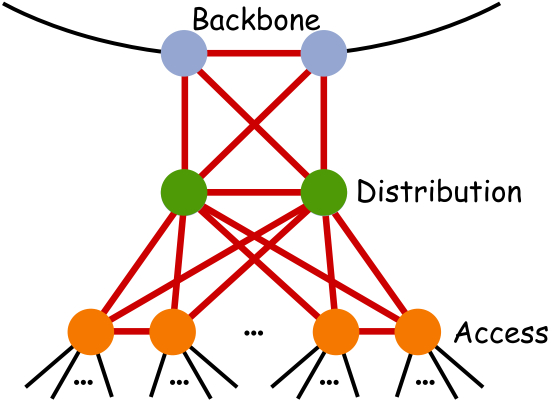
- There can be multiple (or no) factorizations of a graph into different tensor products.
- If either G or H is bipartite then their tensor product is also.
- The tensor product is connected iff both G and H are connected, and at least one factor is non-bipartite.
- Properties derived from those of Kronecker products
  - bilinear
  - associative

Strong product $G * H$

- Defined by
  - $N(G * H) = N(G) \times N(H)$
  - any two vertices (u, u') and (v, v') are adjacent iff
    - $(u', v') \in E(H)$ and $(u, v) \in E(G)$; or
    - $u = v$ and $(u', v') \in E(H)$; or
    - $u' = v'$ and $(u, v) \in E(G)$
  - Its like the union of the Cartesian and Tensor products.

Strong product $G * H$

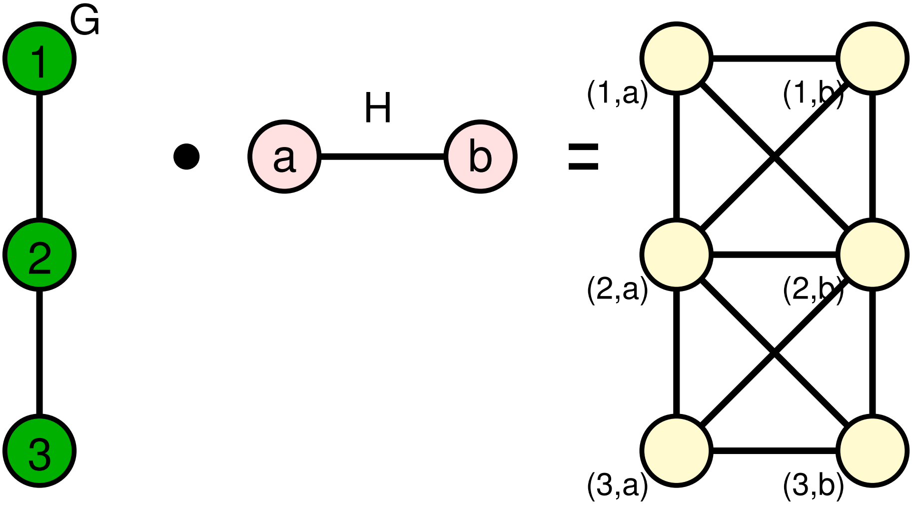
Example network design pattern (within a PoP)



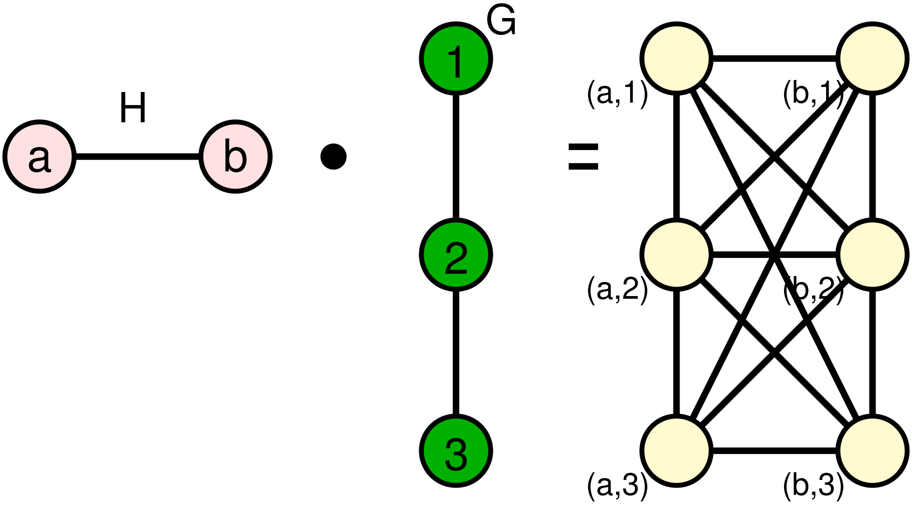
Lexicographic product $G \bullet H$

- Lexicographic product (graph composition) $G \bullet H$
- Defined by
  - $N(G \bullet H) = N(G) \times N(H)$
  - Any two vertices (u, u') and (v, v') are adjacent iff
    - $(u, v) \in E(G)$; or
    - $u = v$ and $(u', v') \in E(H)$
  - This is the first one in which order is really important
    - non-commutative
    - Lexicographic order = dictionary order

Example Lexicographic product



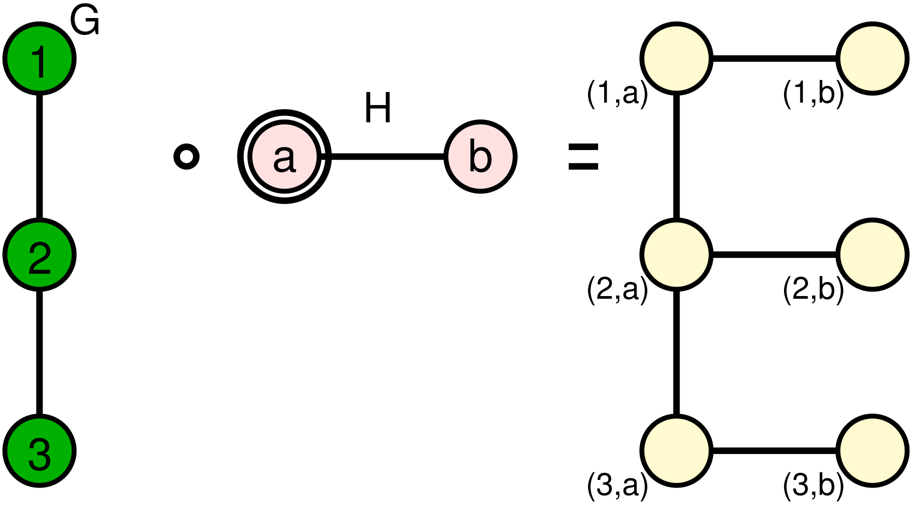
Example Lexicographic product



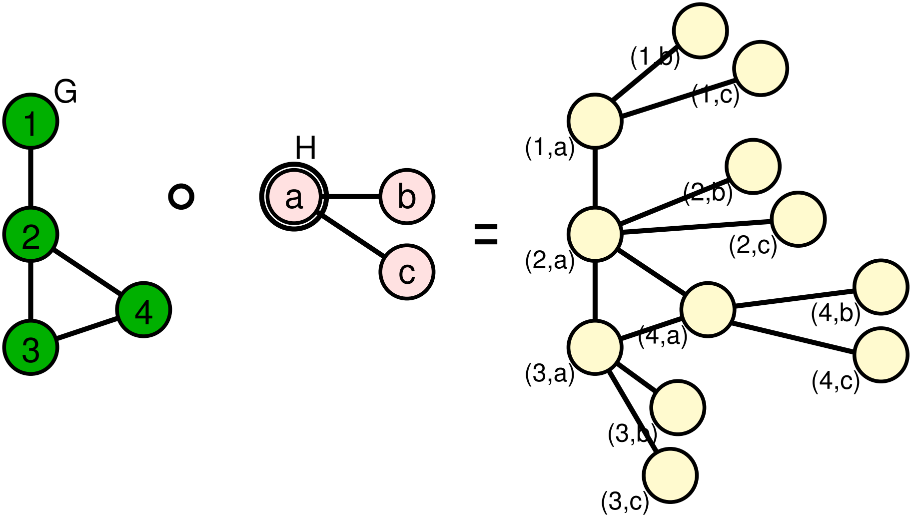
Rooted product $G \circ H$

- Product of G with *rooted* graph H
- Defined by
  - $N(G \circ H) = N(G) \times N(H)$
  - Take the root of H to be $h \in N(H)$
  - Any two vertices (u, u') and (v, v') are adjacent iff
    - $u' = h$ and $v' = h$ and $(u, v) \in E(G)$; or
    - $u = v$ and $(u', v') \in E(H)$
  - Imagine taking $|N(G)|$ copies of H, and associating the root of H with each node of G.

Example Rooted Product



Example Rooted Product



Rooted Product Properties

- Non-commutative
- If G is also rooted then $G \circ H$ is rooted.
- The rooted product of two trees is a tree.

COLD part II

- COLD generated PoP-level map
- Use graph products to construct the layer below
  - multiple-routers as part of PoP
  - multiple links between PoPs (for redundancy)
  - structure inside the PoP

Section 2

Operators on a graph and an edge

Binary Operators on a graph and an edge

- Deletion ($E \leftarrow E \backslash e$)
- Insertion ($E \leftarrow E \cup e$)
- Edge contraction

Edge Contraction

- Merge two adjacent nodes along an edge $e = (u, v)$, $u, v \in N$, $u \neq v$.
- New graphs G', which has
  - nodes $N' = (N \backslash \{u, v\}) \cup \{w\}$
  - edges $E' = E \backslash \{e\}$
  - every edge $(u, i) \in E$ is replaced by $(w, i) \in E'$
    (and the same for links $(v, i) \in E$)

Section 3

Operators on a graph and a node

Binary Operators on a graph and a node

- Deletion
  - remove node n from the graph
  - also delete all edges $(n, i) \in E$ from the graph
- Insertion

Further reading I