A simple approach to computing Assembly Indices

D.G. Moore, S. Marshall, Y. Liu, S.I. Walker, L. Cronin

14 June 2020

Recall that assembly spaces (Γ, ϕ) are defined in the supplementary information as a quiver (multigraph) Γ together with an edge labeling ϕ which satisfies a few axioms. Where possible, we will simply write Γ. The vertices of the assembly space $V(\Gamma)$ represent objects, e.g. strings, molecules, etc..., and edges $E(\Gamma)$ represent a method of constructing that object in a single joining operation from previously constructed (or it a priori) objects. The *a priori* objects are referred to as basic objects, and the set of basic objects is referred to as the *basis of the assembly space*.

The definitions that follow are labeled alphabetically. References to numerically labeled definitions, lemmas and theorems refer to the supplementary information.

Definition a. Let $a, b \in V(\Gamma)$. We write $a \prec b$ if for all $c \in V(\Gamma)$ with $a \leq c \leq b$[1], either $c = a$ or $c = b$.

In the context of an assembly space, $a \prec b$ denotes that b can be constructed from a in a single step by joining a with something else in the space.

Definition b. Let $a \in V(\Gamma)$, the $\coprod a = \{(u, \phi(u)) \in V(\Gamma) \times V(\Gamma) \mid u \prec a\}$. For $S \subseteq V(\Gamma)$, let $\coprod S = \bigotimes_{a \in S} \coprod a$.

You can think of $\coprod a$ as the set of all pairs of things that can be joined to make a. Similarly, $\coprod S$ where S is a set of objects can be thought of as all of the ways that every object in S can be constructed in a single step.

Definition c. Let $U \subseteq V(\Gamma)$. $S(U) = \{a \in U \mid a \in \downarrow(U \setminus \{a\})\}$.

As we are building up objects via pairwise joining operations, we need to be able to identify "shortcuts" when they are possible. For example, consider making $aabaa$. One possible pairwise joining would be $aab * aa$. But notice that $aa \leq aab$. When we then move on to make aab, we want to consider make sure that if we decided to make aab as $aa * b$ we don't double count aa – we already made it in the previous step.

In other words, $S(U)$ determined which elements of U could be used to make something else in U.

[1] See (**Lemma. 1**) for an explanation of $\leq$.

A recursive expression for (co)assembly indicies

As simply as I can put it...

The assembly index of an object $c(x)$ is 1 greater than the smallest coassembly of any pair of objects which can be joint to create x.

The coassembly index $cc(\sigma, \tau)$ of a set of objects σ contingent on a set of "shortcuts" τ, is the number of complex (non-basic) objects in σ plus the smallest coassembly of all objects which can be pairwise joined to make every object in σ.

I'm not quite satisfied with this way of writing things, but it's about all I've got at the moment.

$$\begin{aligned} c(x) &= cc(\{x\}, \emptyset) \\ cc(\sigma, \tau) &= |S \setminus \Gamma_B| + \min_{\alpha \in \coprod \sigma} cc(J(\alpha) \setminus (\Gamma_B \cup S(J(\alpha)) \cup \tau),\ R(J(\alpha)) \cup \tau) \end{aligned}$$

where

$$J(U) = \bigcup_{(a_1,\ldots,a_n) \in U} \{a_1, \ldots, a_n\}.$$

You can find an implementation of this for strings at
`https://github.com/dglmoore/Pathways.jl/blob/master/src/assembly.jl`.