CO 330, LECTURE 7 SUMMARY

FALL 2017

Summary

Today was about combinatorial specifications.

Definition 1. *Let Θ take as input m combinatorial classes $\mathcal{B}^{(1)}, \mathcal{B}^{(2)}, \ldots, \mathcal{B}^{(m)}$ and build as output a combinatorial class $\mathcal{A} = \Theta(\mathcal{B}^{(1)}, \mathcal{B}^{(2)}, \ldots, \mathcal{B}^{(m)})$. Then Θ is* admissible *if for each n there exists an N such that the counting sequence of $\mathcal{A}$ up to n depends only on the counting sequences of the $B^{(i)}$ up to N.*

Note that this is the combinatorial class analogue of valid operations on formal power series. Similarly to the formal power series case we can also be interested in operations that are only admissible on combinatorial classes with some restriction (most commonly, no elements of size 0).

Definition 2. *A* combinatorial specification *for the classes $\mathcal{A}^{(1)}, \mathcal{A}^{(2)}, \ldots, \mathcal{A}^{(r)}$ is a system of equations*

$$\begin{aligned}
\mathcal{A}^{(1)} &= \Theta_1(\mathcal{A}^{(1)}, \mathcal{A}^{(2)}, \ldots, \mathcal{A}^{(r)}) \\
\mathcal{A}^{(2)} &= \Theta_2(\mathcal{A}^{(1)}, \mathcal{A}^{(2)}, \ldots, \mathcal{A}^{(r)}) \\
&\vdots \\
\mathcal{A}^{(r)} &= \Theta_r(\mathcal{A}^{(1)}, \mathcal{A}^{(2)}, \ldots, \mathcal{A}^{(r)})
\end{aligned}$$

where the Θ_i are built from admissible operators along with $\mathcal{Z}$ and $\mathcal{E}$.

We've seen many examples of this so far, and today we did an example which was a system.

There is one lacuna to note here. For ordered rooted trees, should we write $\mathcal{T} = \mathcal{Z} \times \mathrm{Seq}(\mathcal{T})$ or do we need to define Θ as the operator which builds a new tree with its first argument as the root and its later arguments as the subtrees at the root, and then write $\mathcal{T} = \Theta(\mathcal{Z}, \mathrm{Seq}(\mathcal{T}))$? Really we should do the latter, but that makes the notation rather heavy without any real benefit, so we just take the bijection between a tree and the list of its root and the subtrees at the root as understood and write $=$ even when we're using that bijection.

Some definitions about specifications are the following

Definition 3. *Let*

$$\begin{aligned}
\mathcal{A}^{(1)} &= \Theta_1(\mathcal{A}^{(1)}, \mathcal{A}^{(2)}, \ldots, \mathcal{A}^{(r)}) \\
\mathcal{A}^{(2)} &= \Theta_2(\mathcal{A}^{(1)}, \mathcal{A}^{(2)}, \ldots, \mathcal{A}^{(r)}) \\
&\vdots \\
\mathcal{A}^{(r)} &= \Theta_r(\mathcal{A}^{(1)}, \mathcal{A}^{(2)}, \ldots, \mathcal{A}^{(r)})
\end{aligned}$$

be a combinatorial specification then the dependency digraph *of the specification is the directed graph with vertices* $1, 2, \ldots, r$ *and a directed edge from* i *to* j *if* $\mathcal{A}^{(i)}$ *appears on the right hand side of the* j*th equation.*

If a specification has an oriented cycle in its dependency digraph then we say the specification is *recursive*, otherwise it is *iterative*. (Another important question is not just the existence of an oriented cycle but whether or not the dependency digraph is strongly connected, but we didn't talk about that.)

Definition 4. *An iterative specification that involves only disjoint union,* $\times$*,* $\mathcal{E}$*,* $\mathcal{Z}$*, and* Seq *is called a* regular specification.

This is similar to but not the same as the definition of a regular language from computer science. The difference is that for a regular language in CS we only require an iterative expression using those pieces which generates every word; we don't require the words to be generated unambiguously. However, it turns out that if the class is a regular language in the sense of CS then there is a (potentially more complicated) specification which fixes the ambiguities, so the class has a regular specification as well. See the reference below.

Finally, you thought about some examples and I showed you combstruct in maple. There's a file of combstruct examples you can try out on the website along with the link to this lecture summary.

References

For the relation between regular languages and regular specifications see Flajolet and Sedgewick "Analytic Combinatorics" appendix A.7.