DNA Specifications and Codon: Expressing Combinatorial Designs at Scale

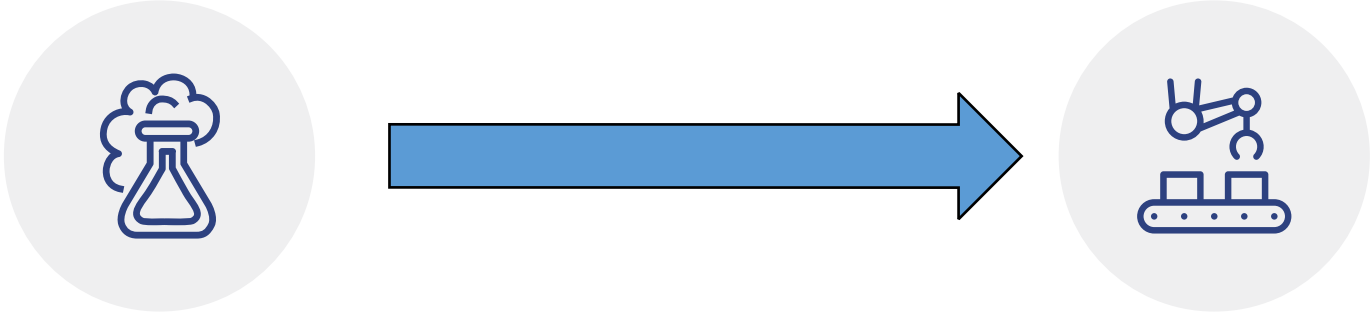
Zach Palchick

zymergen


zymergen

About Zymergen

zymergen



Designers/
Development Scientists

The Factory

What's the problem

1. **Can express large design spaces**
2. **Compact format**
3. **Transferable**
4. **Structure preserving**

DNA Specifications

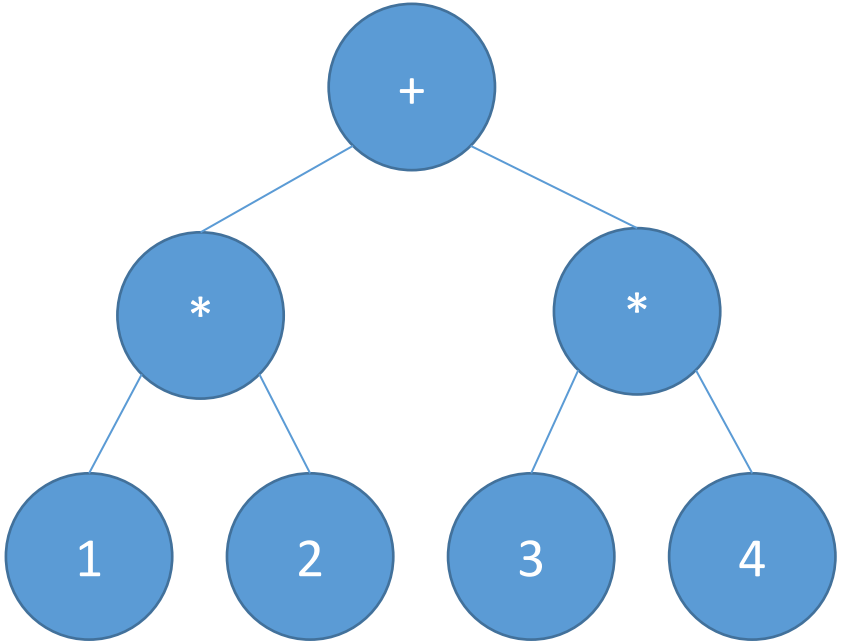
1. **Nested trees of functions**
2. **Leaves are collections of DNA sequences**
3. **Internals are string manipulating operations**
4. **Similar to an Abstract Syntax Tree**

Abstract syntax trees 101

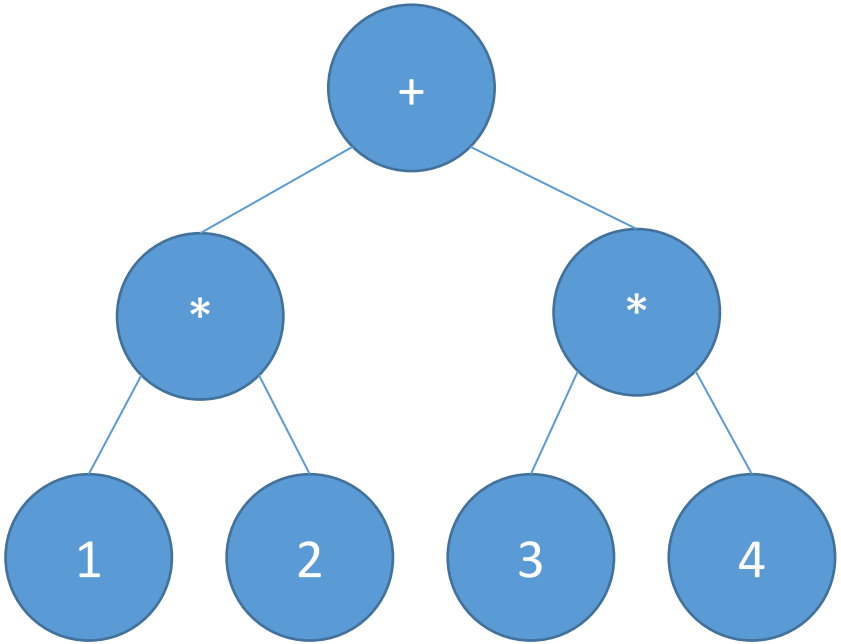
x = (1 * 2) + (3 * 4)
x = (2) + (3 * 4)
x = (2) + (12)
x = 2 + (12)
x = 2 + 12
x = 14

Algebra can be modeled as an AST

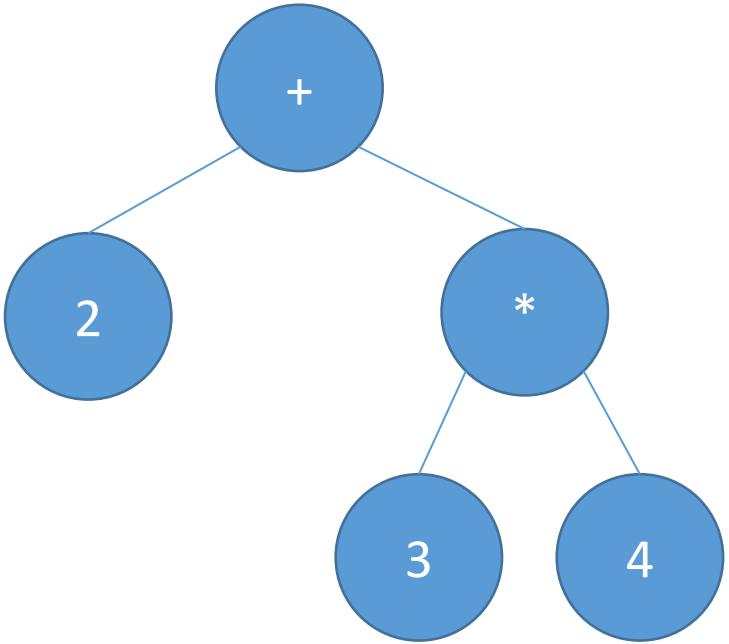
x = (1 * 2) + (3 * 4)



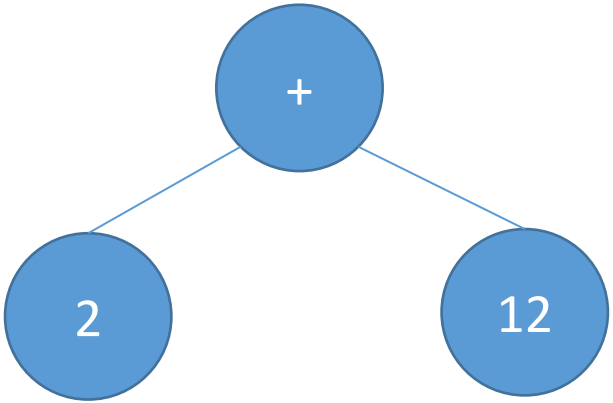
Evaluating ASTs



Evaluating ASTs

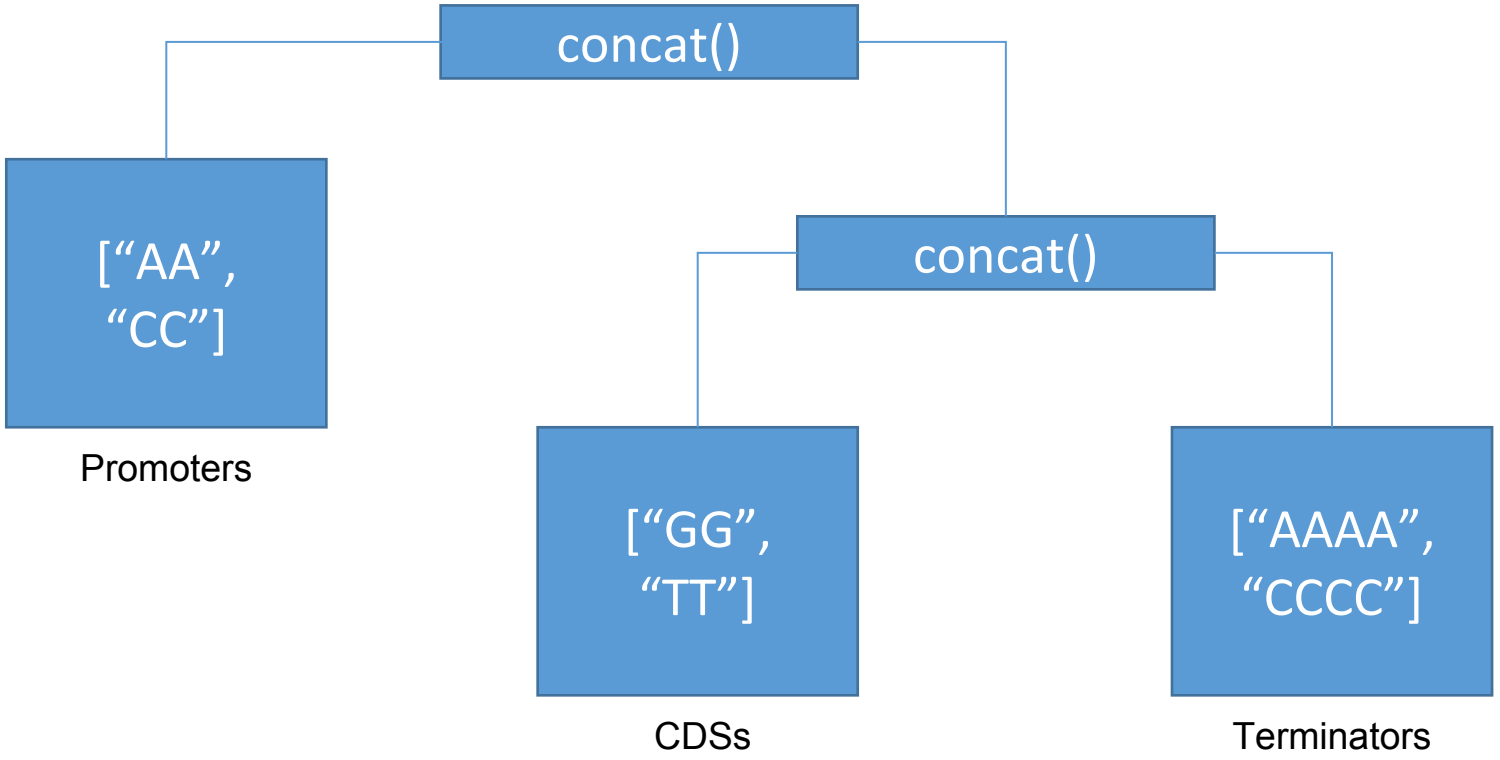


Evaluating ASTs

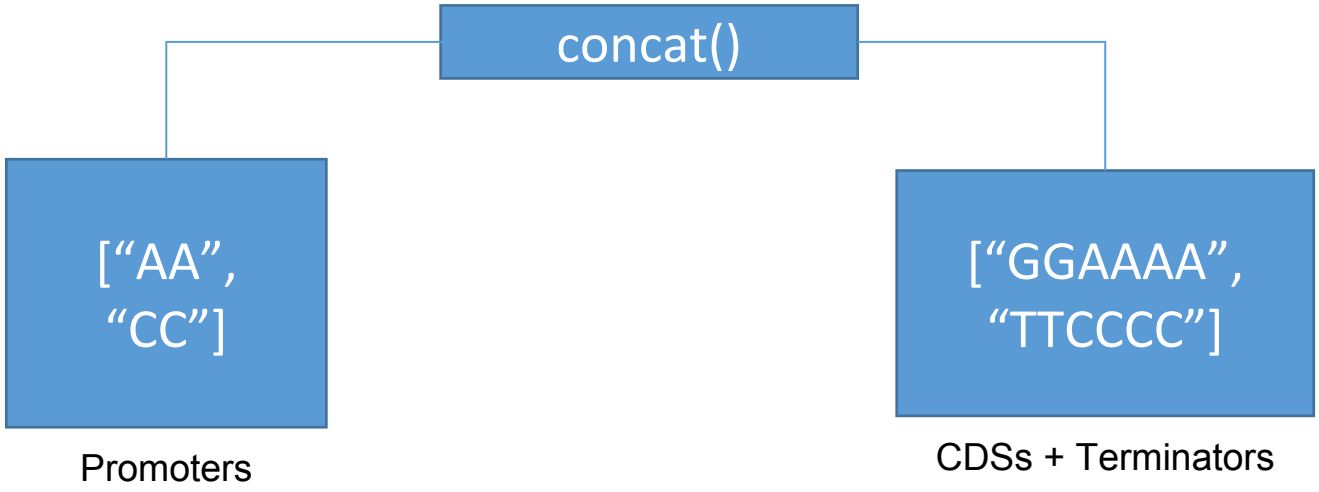


Evaluating ASTs

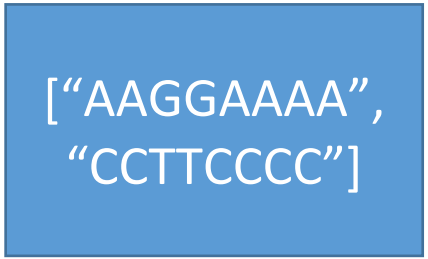
DNA Specifications are ASTs



DNA Specifications are ASTs

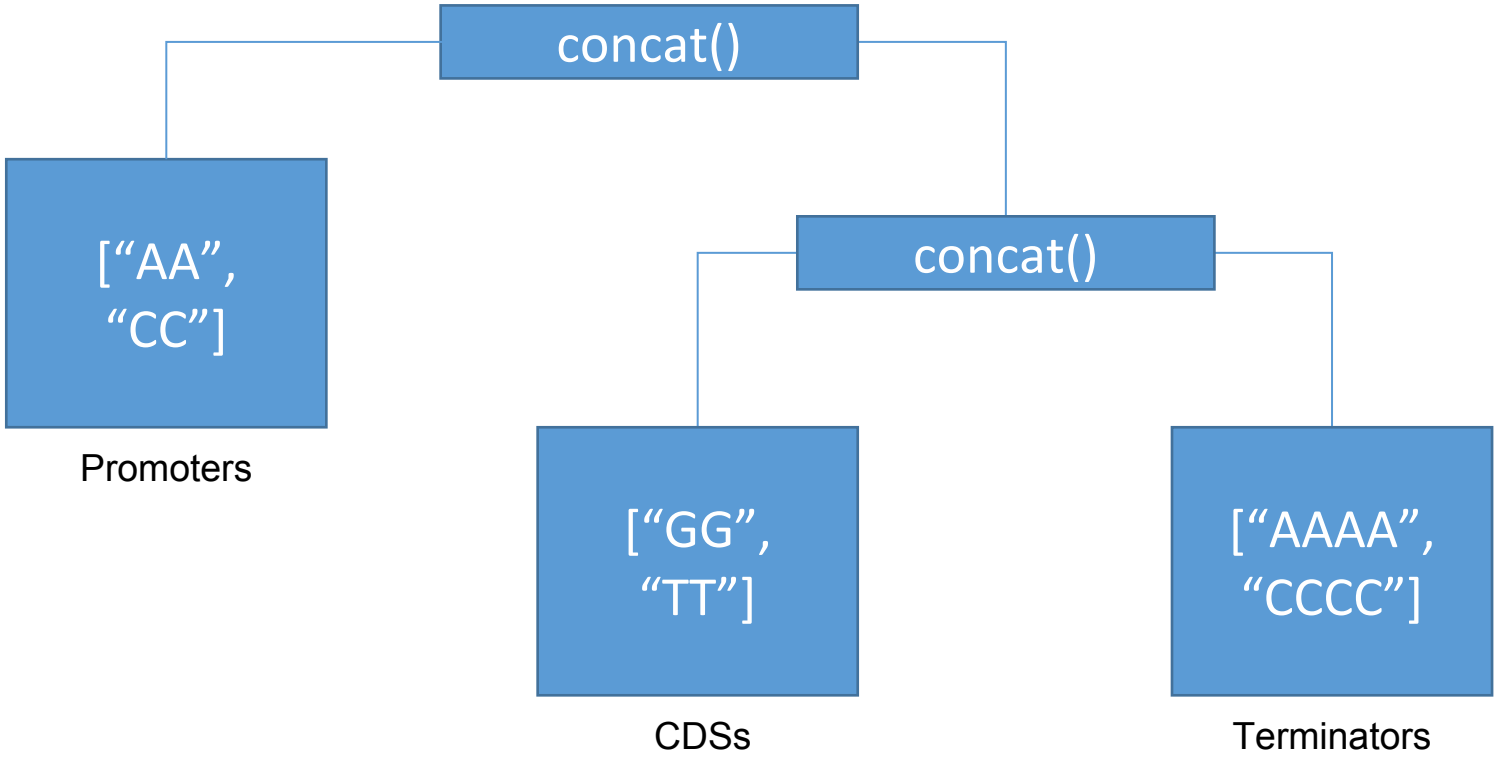


DNA Specifications are ASTs

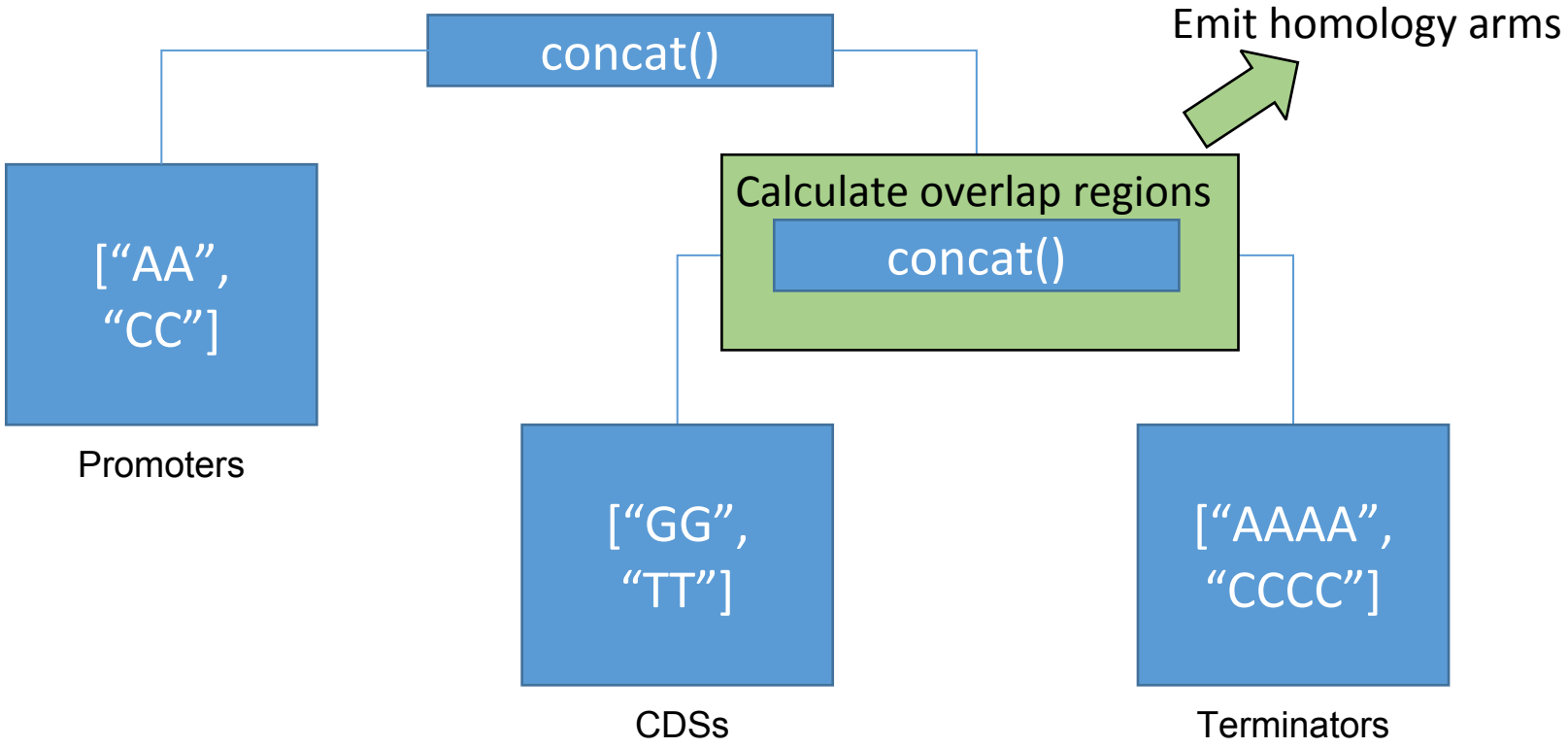


Promoters + CDSs + Terminators

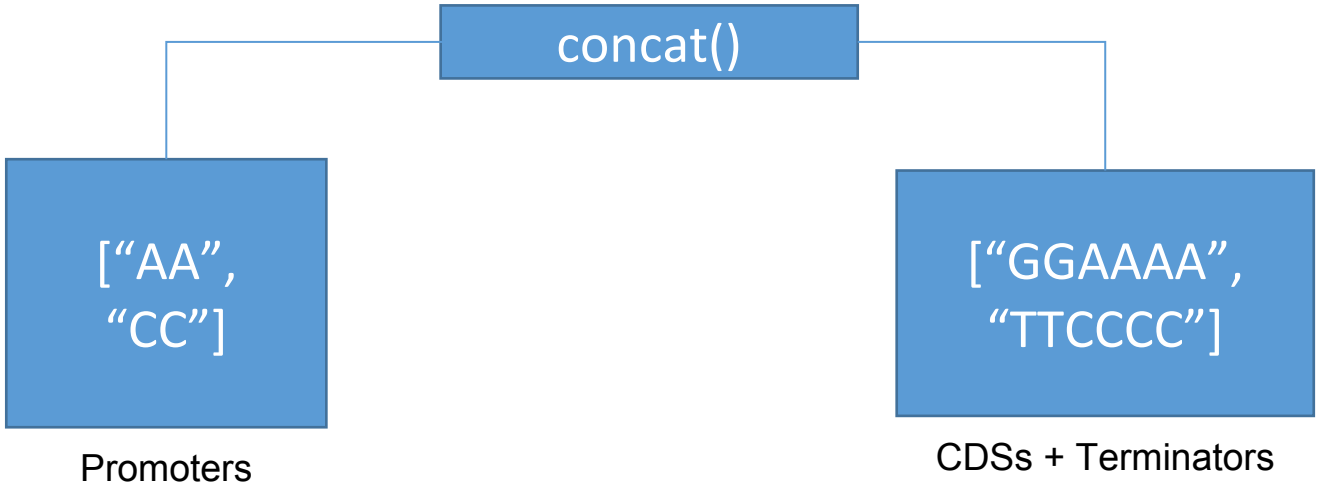
ASTs let us preprocess structure



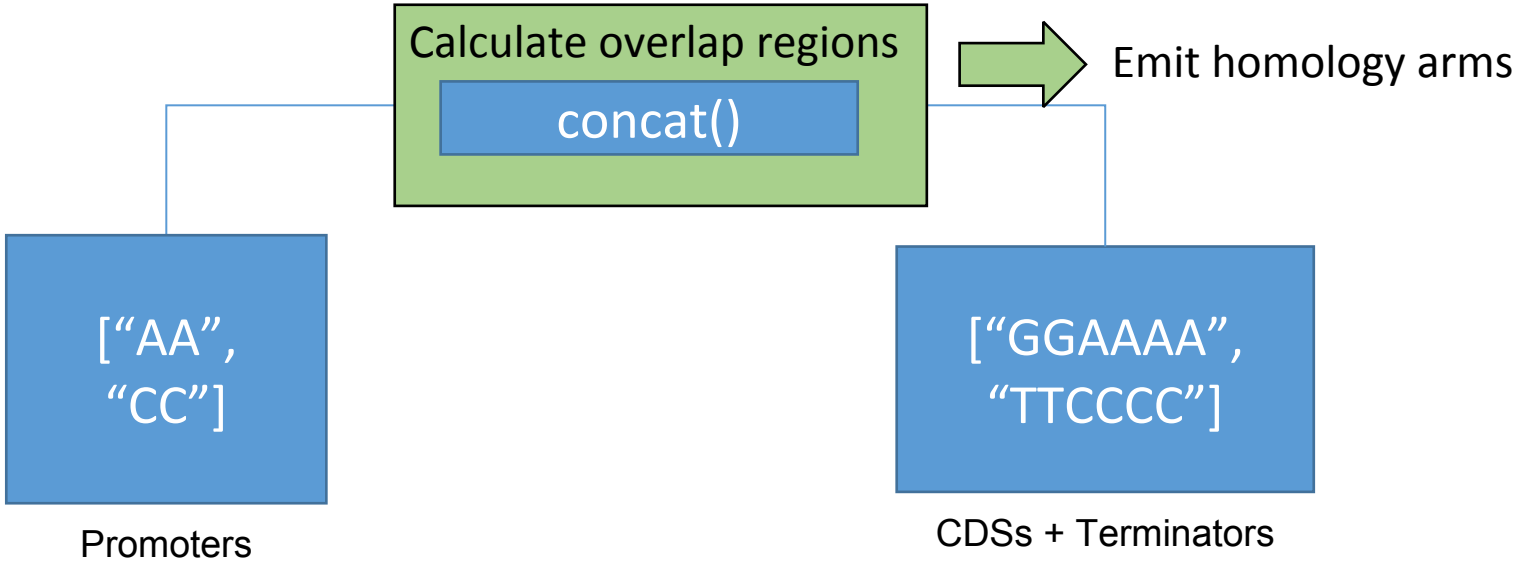
ASTs let us preprocess structure



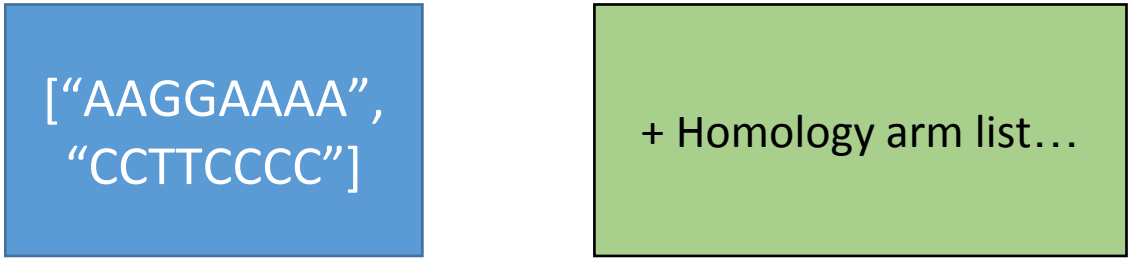
ASTs let us preprocess structure



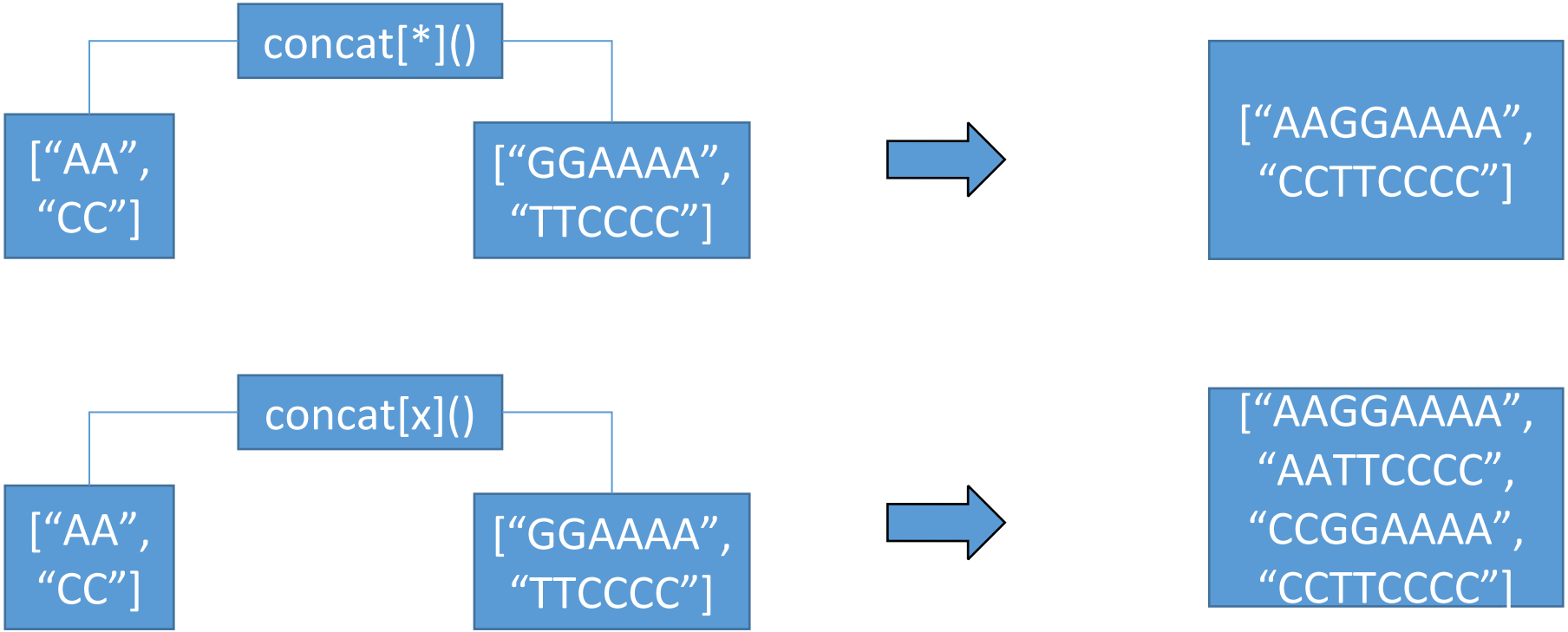
ASTs let us preprocess structure



ASTs let us preprocess structure



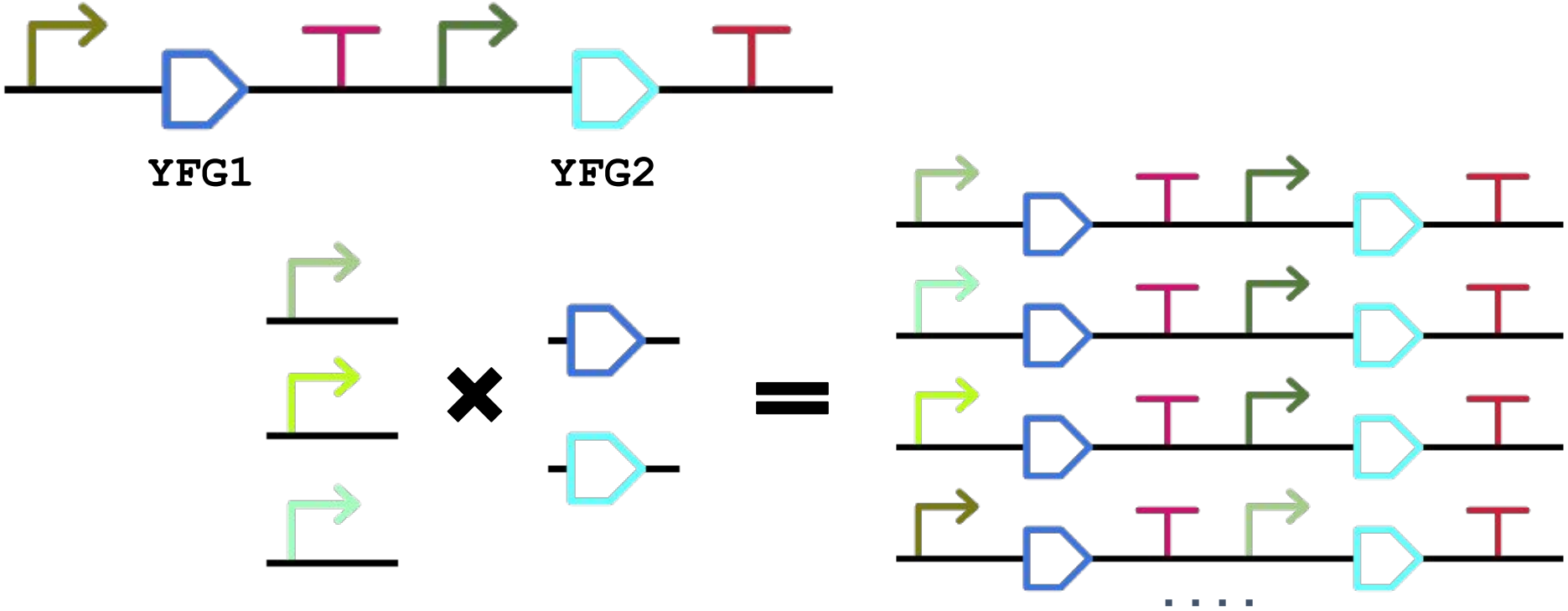
Functions can be parametric



Codon: Promoter Swapping

```
replacePromoter[x](locateGenes(hostStrain, "YFG*"), promoters)
```

Codon: Promoter Swapping

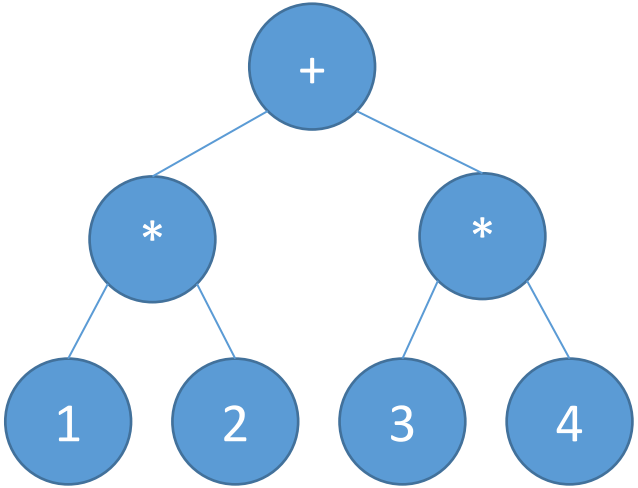



```
replacePromoter[x](locateGenes(hostStrain, "YFG*"), promoters)
```

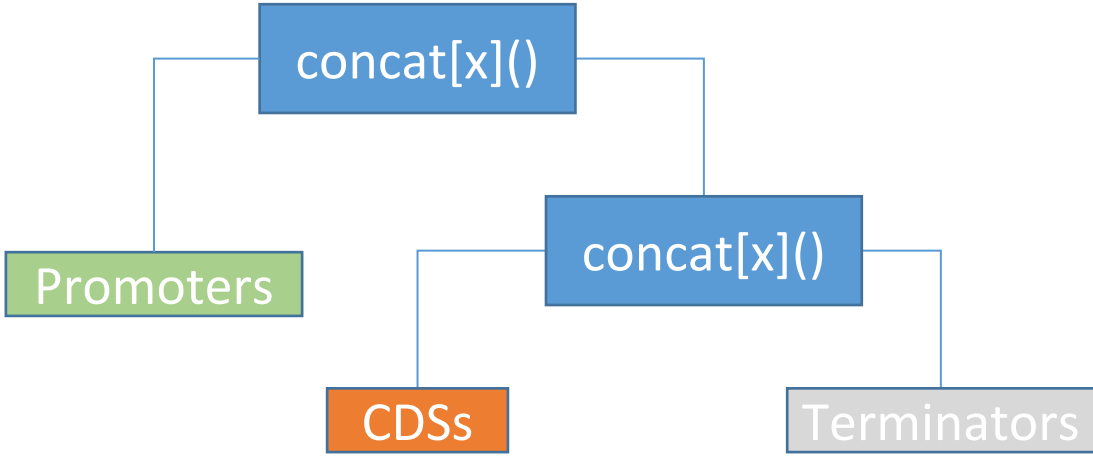
How well do DNA Specifications work?

Demonstration of mapping

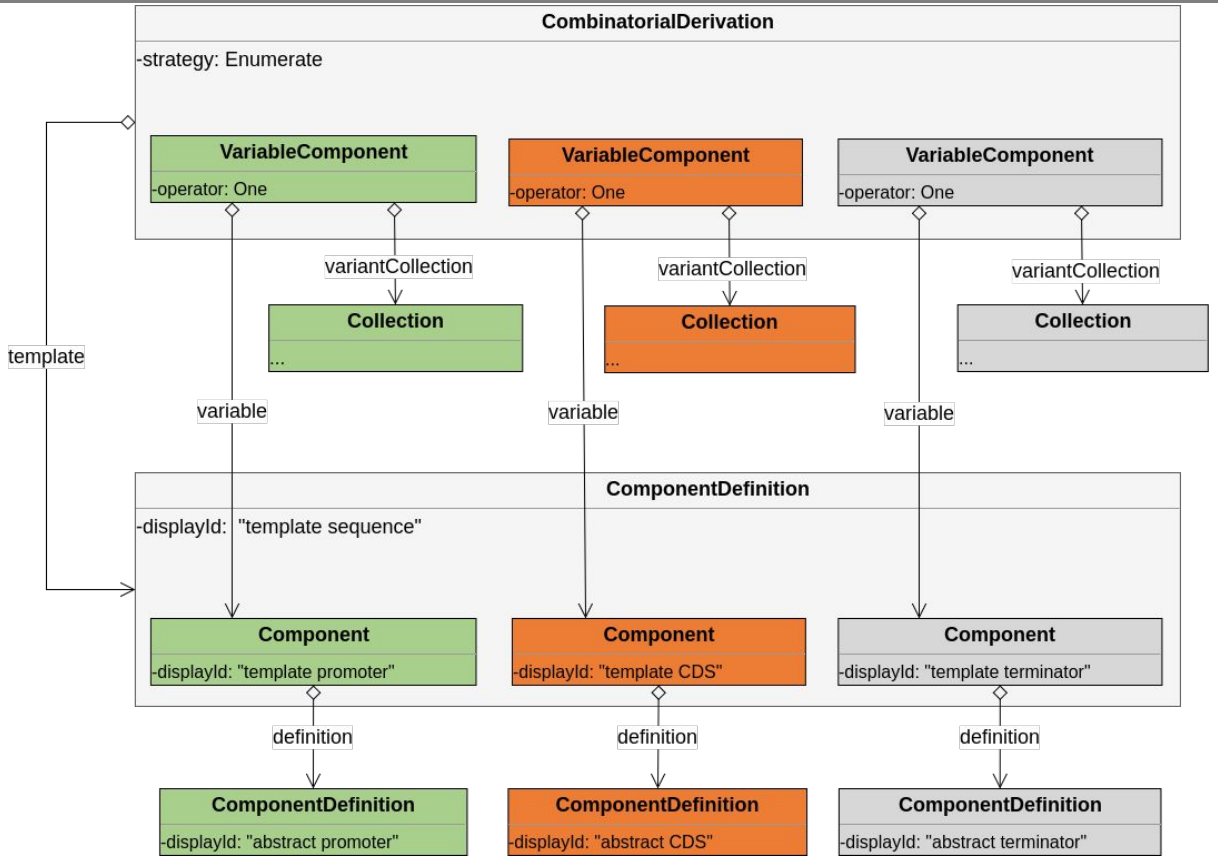
x = (1 * 2) + (3 * 4)



Promoters x CDSs x Terminators

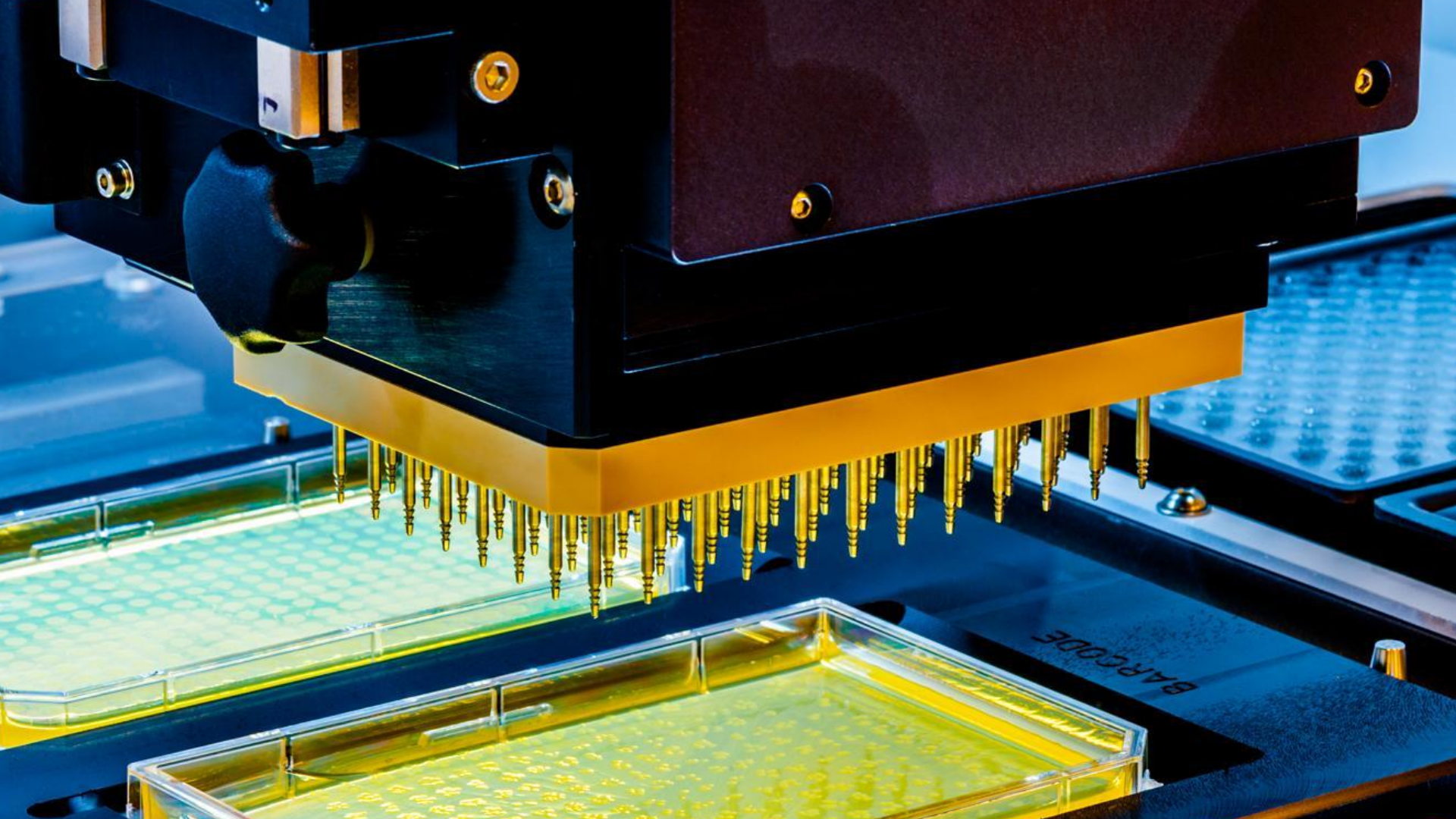


Demonstration of mapping



Thank You

Zach Palchick - palchicz@zymergen.com


BARCODE