**Question**: How can program synthesis help a core science?

What are the opportunities for applying synthesis to the production of 1) life-saving therapeutic drugs, 2) green non-fossil fuels, 3) green polymers materials?
Our thrust: Suggesting/inferring/synthesizing new series of reactions (biochemical pathways) for the bioengineer to “insert” into a bacteria/yeast.

Outcome: We suggest pathways; engineer a strain of bacteria with pathway; bacteria feeds on sugars/biomass to produce target chemicals

Targets: 1) arbekacin-- a last resort antibacterial, 2) butanol-- a alternative to gasoline, 3) precursors to biodegradable polymers
Enzymes are proteins that catalyze reactions.

Reactions involve chemicals; which can be viewed as connected graphs (nodes are atoms; edges bonds).

Each reaction can be viewed as transforming a set input graphs (chemicals) to output graphs.
These graph transforms are really operators (biologists call enzymes promiscuous as they operate over a range of “similar” input chemicals)

Looking at many transforms, we can infer operators

Transforms are operator applications
In fact, we can define a hierarchy of reaction operators.
In fact, we can define a hierarchy of reaction operators.

**Hierarchy of operators**

- Core ROs (Mass-balance)
- Electronic ROs (VSEPR) ...
- Statistical ROs (Seen; stat)
- Fixed ROs (Seen; concretely)
In fact, we can define a hierarchy of reaction operators:

**Hierarchy of operators**

- Core ROs (Mass-balance)
- Electronic ROs (VSEPR)...
- Statistical ROs (Seen; stat)
- Fixed ROs (Seen; concretely)
In fact, we can define a hierarchy of reaction operators

Core ROs (Mass-balance)
Electronic ROs (VSEPR) ...
Statistical ROs (Seen; stat)
Fixed ROs (Seen; concretely)
In fact, we can define a hierarchy of reaction operators

Core ROs (Mass-balance)  
Electronic ROs (VSEPR) ...  
Statistical ROs (Seen; stat)  
Fixed ROs (Seen; concretely)
Find a sequence of operator instantiations $o_1, o_2, \ldots, o_n$ such that:

a) Soundness:
   1) Exist graphs $g_0, g_1, \ldots, g_n$
   2) $g_0 = \text{src molecule}$ and $g_n = \text{dst molecule}$
   3) $o_i$ instantiates some $O_i$; $O_i \in \text{Universe of Ops}$

b) Optimality: Operators $O_i$ are maximally low (closest to naturally seen)
The search space is defined by how permissive the lhs matching pattern of the operator is: Smaller lhs graphs are more permissive.
The search space is defined by how permissive the lhs matching pattern of the operator is: Smaller lhs graphs are more permissive.
The search space is defined by how permissive the lhs matching pattern of the operator is: Smaller lhs graphs are more permissive.

Largest lhs; Search is BFS
Least likely the right transform exists

Synthesis search space
The search space is defined by how permissive the lhs matching pattern of the operator is: Smaller lhs graphs are more permissive.

Smallest lhs; search exponential; greatest exponent
Most likely a transform exists

Largest lhs; Search is BFS
Least likely the right transform exists

Synthesis search space
The search space is defined by how permissive the lhs matching pattern of the operator is: Smaller lhs graphs are more permissive.

- **Universe RO**
  - \( \text{CRO}_1 \)
  - \( \text{CRO}_2 \)
  - \( \text{CRO}_3 \)
  - ... (continued)

  **Smallest lhs; search exponential; greatest exponent**
  - Most likely a transform exists

  **In-between lhs; search exponential**
  - Transform likely exists: \( >\text{RO}, <\text{CRO} \)
  - ... (continued)

  **Largest lhs; Search is BFS**
  - Least likely the right transform exists

---

**Synthesis search space**
Modularity
Modularity
What helps:
a) Requirement for maximally low.
b) Modularity (goes hand in hand with aggregate op).

To think about:
a) Best way to traverse hierarchy: symbolic exploration?
b) Better formulation than graph transformations?
   - Incremental data structure construction?
   - Synthesis of functional programs with composition?
   - Interfaces matching?
Synthesis formulation:
\[ \exists p \ \forall x: \ spec(x, p(x)) \]

\[ \exists \ \text{pathway} \ \forall \ \text{outchem} : \]
\[ \text{pathway}^{-1}(\text{outchem}) \subseteq \text{metabolites} \]

Constraint solving
case most important
Synthesis formulation:

\[ \exists p \ \forall x: \text{spec}(x, p(x)) \]

\[ \exists \text{pathway} \ \forall \text{outchem} : \]
\[ \text{pathway}^{-1}(\text{outchem}) \subseteq \text{metabolites} \]

Of real interest, constraint solving formulation:

\[ \exists \text{pathway} : \text{pathway}^{-1}(\text{outchem}) \subseteq \text{metabolites} \]
Synthesis formulation:
\[ \exists p \ \forall x: \text{spec}(x, p(x)) \]
\[ \exists \text{pathway} \ \forall \text{outchem} : \]
\[ \text{pathway}^{-1}(\text{outchem}) \subseteq \text{metabolites} \]

Of real interest, constraint solving formulation:
\[ \exists \text{pathway} : \text{pathway}^{-1}(\text{outchem}) \subseteq \text{metabolites} \]

Can we add a decision procedure to Z3/CVC?

Constraint solving case most important
DPLL(T):
T lemma: tell solver contradictions (block states)
T propagate: add implications (augment learned)


Optimization problems (min-cost SAT) can be implemented as evolving stronger theory

http://www.lsi.upc.edu/~oliveras/espai/papers/sat06.pdf

Decision procedures

Most difficult part if permutation in molecular graphs; factored out in theory
DPLL(T):
- T lemma: tell solver contradictions (block states)
- T propagate: add implications (augment learned)


Optimization problems (min-cost SAT) can be implemented as evolving stronger theory
http://www.lsi.upc.edu/~oliveras/espai/papers/sat06.pdf

Graphs as Godel numbers: Init set + derived. Numbering follows the pattern of natural derivation.

Transforms: \((a,b,c) \Rightarrow^t (x,y), \ t \in \text{transforms}\)

Theory T: checks \((a,b,c) \Rightarrow^t (x,y)\) is T-consistent, i.e., T lemma about it. No propagate?

Decision procedures

Most difficult part if permutation in molecular graphs; factored out in theory
Formulation:

\[ \exists g^{c_{1..k}}, g^{t_{1..r}} : \bigwedge a_i \supseteq^t a_{i+1} \]

- Mincost sat solution desired; where \( \supseteq \) is defined by the decision procedure for the theory of molecular graphs and transforms
- Allows us to add additional constraints
Formulation:

\[ \exists g^{c_{1..k}}, g^{t_{1..r}} : \bigwedge a_i \Rightarrow^t a_{i+1} \]

- Mincost sat solution desired; where \( \Rightarrow \) is defined by the decision procedure for the theory of molecular graphs and transforms
- Allows us to add additional constraints

But if simpler (if no additional constraints):

\[ \exists g^{c_{1..k}}, g^{t_{1..r}} : \bigwedge a_i \Rightarrow^t a_{i+1} \]

- On demand instantiated traversal?

SMT problem or not?
SMT problem or not?
SMT problem or not?
SMT problem or not?
SMT problem or not?
SMT problem or not?
1) Cost in path vs cost in abstraction
2) Cannot fully enumerate
3) When abstract outside of domain

SMT problem or not?
In perspective:

- Natural transform data
- Derived Operators
- Modularize
- Synthesize Pathway
- Genes (DNA) for Op
- Plasmid to insert
- Functional Bacteria

End-to-end
In perspective:

Natural transform data

Derived Operators

Modularize

Synthesize Pathway

Genes (DNA) for $\text{Op}_i$

Plasmid to insert

Functional Bacteria

End-to-end
In perspective:

Derived Operators

Modularize

Synthesize Pathway

Genes (DNA) for $O_{pi}$

Plasmid to insert

Functional Bacteria

End-to-end
In perspective:

- Natural transform data
- Derived Operators
- Modularize
- Synthesize Pathway
- Genes (DNA) for Op_i
- Plasmid to insert
- Functional Bacteria

End-to-end
In perspective:

Natural transform data
Derived Operators
Modularize
Synthesize Pathway
Genes (DNA) for $O_{p_i}$
Plasmid to insert
Functional Bacteria

End-to-end
In perspective:

Natural transform data
Derived Operators
Modularize
Synthesize Pathway
Genes (DNA) for $O_{p_i}$
Plasmid to insert
Functional Bacteria

End-to-end
End-to-end

In perspective:

Natural transform data
Derived Operators
Modularize
Synthesize Pathway
Genes (DNA) for Op_i
Plasmid to insert
Functional Bacteria

\[
g_0 \xrightarrow{\text{op}_0} g_1 \xrightarrow{\text{op}_1} g_2 \xrightarrow{\text{op}_2} \ldots \xrightarrow{\text{op}_{n-1}} g_n
\]

\(
\{ \ldots \}
\)

biofuel
therapeutic drug
In perspective:

Natural transform data
Derived Operators
Modularize
Synthesize Pathway
Genes (DNA) for Op
Plasmid to insert
Functional Bacteria

BioE God proposes
{ , , , , }

Lesser Gods Optimize

End-to-end
In perspective:

- Natural transform data
  - Derived Operators
    - Modularize
  - Synthesize Pathway
  - Genes (DNA) for $O_{pi}$
    - Plasmid to insert
  - Functional Bacteria

End-to-end

BioE God proposes

{ , , , , }
In perspective:

- Natural transform data
- Derived Operators
- Modularize
- Synthesize Pathway
- Genes (DNA) for Op_i
- Plasmid to insert
- Functional Bacteria

Automated synthesizer over mined operators

End-to-end

Lesser Gods Optimize
**Pathway Synthesis:** Synthesizing novel bio-fuels pathways

**Model Synthesis:** Infer concurrent models that explain experiments

Biological processes as programs

**Synbio Verifier:** Syntactic & semantic verifier for plasmid DNA

Safety analyses for bug-free wet-lab experiments

Other projects

Saurabh + Ras, Sanjit, Chris. Students: Jon, Paul, Jeff, Stephi, Tim, Andrew, Ali, Evan