TASBE: A Tool-Chain for Accelerating Synthetic Biology Engineering


3rd Int’l Workshop on Bio-Design Automation
June, 2011

Work sponsored by DARPA I2O under contract HR0011-10-C-0168; the views and conclusions contained in this document are those of the authors and not DARPA or the U.S. Government.
Vision: WYSIWYG Synthetic Biology

Bioengineering should be like document preparation:
Vision: WYSIWYG Synthetic Biology

Bioengineering should be like document preparation:
Why is this important?

- Breaking the complexity barrier:
  - Multiplication of research impact
  - Reduction of barriers to entry

*Sampling of systems in publications with experimental circuits*
Why a tool-chain?

This gap is too big to cross with a single method!
The TASBE architecture:

Organism Level Description

High Level Description

Abstract Genetic Regulatory Network

DNA Parts Sequence

Assembly Instructions

Cells

Modular architecture also open for flexible choice of organisms, protocols, methods, ...
A Tool-Chain Example

(yellow (not (cyan (Dox))))
A Tool-Chain Example

(yellow (not (cyan (Dox))))

Dox → cyan → not → yellow
A Tool-Chain Example

(yellow (not (cyan (Dox))))

Dox → cyan → not → yellow
A Tool-Chain Example

(yellow (not (cyan (Dox))))

Organism Level Description

High level simulator

Coarse chemical simulator

Abstract Genetic Regulatory Network

Detailed chemical simulator

DNA Parts Sequence

Assembly Instructions

Testing

Cells
A Tool-Chain Example

(yellow (not (cyan (Dox))))
A Tool-Chain Example

(yellow (not (cyan (Dox))))

Organism Level Description
High level simulator

High Level Description
Coarse chemical simulator

Abstract Genetic Regulatory Network
Detailed chemical simulator

DNA Parts Sequence

Assembly Instructions

Testing

Cells

Assemble

<table>
<thead>
<tr>
<th>Part</th>
<th>Quantity</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>pHef1a</td>
<td>20 µl</td>
<td>Training Work (Col. 1, Row 2)</td>
</tr>
<tr>
<td>rtTA</td>
<td>20 µl</td>
<td>Water-free dispense Work (Col. 2, Row 2)</td>
</tr>
<tr>
<td>pTre</td>
<td>20 µl</td>
<td>Training Work (Col. 1, Row 3)</td>
</tr>
<tr>
<td>CFP</td>
<td>20 µl</td>
<td>Water-free dispense Work (Col. 1, Row 1)</td>
</tr>
</tbody>
</table>
Current state of the tool-chain:

- End-to-end software integration
- Automated designs match hand-generated systems verified *in vivo*
- Some protocols automated

Next: verification of automated design & assembly
Advances on Two Key Problems:

Organism Level Description

High Level Description

Abstract Genetic Regulatory Network

DNA Parts Sequence

Assembly Instructions

Cells

Compilation & Optimization

Characterization of Transfer Curves

If detect explosives: emit signal
If signal > threshold: glow red
Advances on Two Key Problems:

Organism Level Description

High Level Description

Abstract Genetic Regulatory Network

DNA Parts Sequence

Assembly Instructions

Cells

Compilation & Optimization

High level simulator

Coarse chemical simulator

Detailed chemical simulator

Testing

If detect explosives: emit signal
If signal > threshold: glow red
Transcriptional Logic

Decay $\rightarrow$ Protein

Signal = Concentration

Stabilizes at $\text{decay} = \text{production}$
BioCompiler Overview

- BioCompiler converts high level program to abstract GRN
- Motifs map high level operators to parameterized implementation in biology.
- As with all compilers, the initial mapping can be greatly optimized.
Motif-Based Compilation

• High-level primitives map to GRN design motifs
  – e.g. logical operators:

  \[(\text{primitive not (boolean) boolean :grn-motif ((P high R- arg0 value T)))}\]
Motif-Based Compilation

- High-level primitives map to GRN design motifs
  - e.g. logical operators, **actuators**:

```
(primitive green (boolean) boolean :side-effect
 :type-constraints ((= value arg0))
 :grn-motif ((P R+ arg0 GFP|arg0 value T)))
```
Motif-Based Compilation

- High-level primitives map to GRN design motifs
  - e.g. logical operators, actuators, sensors:

\[
\text{(primitive IPTG () boolean}
\quad :\text{grn-motif ((P high LacI|boolean T)}
\quad \text{(RXN (IPTG|boolean) represses LacI)}
\quad \text{(P high R- LacI value T))}
\]
Motif-Based Compilation

- Functional program gives dataflow computation:

  \[(\text{green } (\text{not } (\text{IPTG})))\]
Motif-Based Compilation

- Functional program gives dataflow computation:

\[(\text{green } (\text{not } (\text{IPTG})))\]
Motif-Based Compilation

- Operators translated to motifs:
Motif-Based Compilation

- Operators translated to motifs:

```
IPTG  not  green
```

```
LacI  outputs
```

```
IPTG  A  arg0  outputs
```

```
B  arg0  GFP  outputs
```
Motif-Based Compilation

- Operators translated to motifs:

```
IPTG   not   green

LacI    outputs  arg0

IPTG

LacI    A

B

GFP  outputs
```
Optimization

LacI

IPTG

A

B

GFP
Optimization

Copy Propagation
Optimization

Copy Propagation

Dead Code Elimination
Optimization

- Copy Propagation
- Dead Code Elimination
- Dead Code Elimination
Complex System: Feedback Latch

(def \texttt{sr-latch} (s r)
   (letfed+ ((o boolean (not (or r o-bar)))
             (o-bar boolean (not (or s o))))
     o))

(green \texttt{(sr-latch \texttt{aTc} \texttt{IPTG}))))
Complex System: Feedback Latch

(letfed+ ((o boolean (not (or r o-bar))))
  (o-bar boolean (not (or s o))))

(green (sr-latch (aTc) (IPTG)))

Unoptimized: 15 functional units, 13 transcription factors
Optimization of Complex Designs

(def sr-latch (s r)
  (letfedd+ ((o boolean (not (or r o-bar)))
    (o-bar boolean (not (or s o))))
  o))

(green (sr-latch (aTc) (IPTG)))
Optimization of Complex Designs

(def sr-latch (s r)
  (letf+ ((o boolean (not (or r o-bar)))
                      (o-bar boolean (not (or s o)))
                      o))

(green (sr-latch (aTc) (IPTG)))

Unoptimized: 15 functional units, 13 transcription factors

Common Subexp. Elim.
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
     (o-bar boolean (not (or s o))))
  o))

(green (sr-latch (aTc) (IPTG)))

Unoptimized: 15 functional units, 13 transcription factors

NOR Compression
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
     (o-bar boolean (not (or s o))))
    o))

(green (sr-latch (aTc) (IPTG)))
Optimization of Complex Designs

(defun sr-latch (s r)
  (letfished+ ((o boolean (not (or r o-bar))))
    (o-bar boolean (not (or s o)))))

(green (sr-latch (aTc) (IPTG)))

Unoptimized: 15 functional units, 13 transcription factors
Optimization of Complex Designs

(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
             (o-bar boolean (not (or s o))))
    o))

green (sr-latch (aTc) (IPTG))

Unoptimized: 15 functional units, 13 transcription factors

Common Subexp. Elim.
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
    (o-bar boolean (not (or s o))))
     o))

(green (sr-latch (aTc) (IPTG)))

Optimization of Complex Designs

Unoptimized: 15 functional units, 13 transcription factors

Dead Code Elimination
Optimization of Complex Designs

```
(def sr-latch (s r)
  (letf'd+ ((o boolean (not (or r o-bar)))
          (o-bar boolean (not (or s o))))
    o))

(green (sr-latch (aTc) (IPTG))))
```

Unoptimized: 15 functional units, 13 transcription factors
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
    (o-bar boolean (not (or s o)))))
  o))

(green (sr-latch (aTc) (IPTG)))

Unoptimized: 15 functional units, 13 transcription factors
Optimization of Complex Designs

```
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
            (o-bar boolean (not (or s o))))
    o))

(green (sr-latch (aTc) (IPTG)))
```

Unoptimized: 15 functional units, 13 transcription factors

Dead Code Elimination
Optimization of Complex Designs

(def sr-latch (s r)
  (letfeder+ ((o boolean (not (or r o-bar))))
    (o-bar boolean (not (or s o))))

(green (sr-latch (aTc) (IPTG)))

Final Optimized:
5 functional units
4 transcription factors

Unoptimized: 15 functional units, 13 transcription factors
Compilation & Optimization Results:

- Automated GRN design for arbitrary boolean logic and feedback systems
- Optimization competitive with human experts:
  - Test systems have 25% to 71% complexity reduction
  - Optimized systems homologous with hand design
Advances on Two Key Problems:

Organism Level Description

High level simulator

High Level Description

Coarse chemical simulator

Abstract Genetic Regulatory Network

Detailed chemical simulator

DNA Parts Sequence

Assembly Instructions

Testing

Cells

Characterization of Transfer Curves
Key Problem: Device Characterization

Goal: quantify single-cell I/O concentration relation

Three phases of lab work:
- Multi-plasmid (qualitative test)
- Single-plasmid (rough quantitative)
- Chromosomal integration (fine quantitative)
Example System: C7C7 repressor
Example System: VP16Gal4 activator

```
Example System:
VP16Gal4 activator

```

![Diagram](image)
From Fluorescence to Static Discipline

Fluorescence of proxy proteins at N hours

Color-corrected Fluorescence

Model-Compensated Transfer Curve

Inflection Points

Fluorescence of EYFP on AmCyan measurements

Repressor→output transfer curve w. varying plasmid count
Color Correction: Bleed-over Matrix

- Selecting an optimal combination of proteins:

![Color Correction Matrix Diagram](image-url)
Color Correction: Piecewise Models

EYFP impact on AmCyan measurements

- Data
- Linear fit (midrange)
- Linear fit (all)
Color Correction: Piecewise Models

EYFP impact on AmCyan compensation models

- Piecewise fit
- 2 std. dev.
- Linear fit (all)
Characterization Contributions

- Multi-stage characterization protocol for transcriptional devices
- Model mapping multi-plasmid behavior to predictions of single copy behavior
- Improved FACS color correction
- Preliminary characterization results
Contributions:

- TASBE: open tool-chain architecture
- Demonstration of end-to-end automated design
- Advances on key sub-problems:
  - Compilation and Optimization
  - DNA Part Selection [Next talk]
  - Flexible Protocol Automation [Following talk]
  - Characterization of Transfer Curves
Toward a community platform…

- Free, open source core
  - Proto, Clotho available now, others by arrangement
- Work on interchange standards (SBOL, CHRIS)