The AutoBio Instruction Set Manual

2022

Preface

This is version 0.1 (draft) of the document describing the AutoBio Instruction Set Architecture (AutoBio ISA). The purpose of the Instruction Set is to provide an interface between humans and machines in the field of biology, i.e., to formulate a clearly defined exchangeable format for biology experiments.

Our aim is to use AutoBio ISA as an alternative to biology protocols in natural languages, and this instruction set is specially designed for experiment standardization.

Currently, this instruction set is far from finished, refinement is welcomed. For more information, please visit

`https://www.overleaf.com/6629364199jbrwrrxyyrqg`

Inhaltsverzeichnis

1 Instruction Overview

The AutoBio-ISA is a new instruction set architecture(ISA) designed to support computer-aided biological experimental automation. Our goals in defining AutoBio-ISA include the following:

- A *real* ISA suitable for biological research, not just limited to computer science.
- An ISA that avoids *"over-natural"* in the current protocol standard. We aim to make each step as clear as possible.
- An ISA that *omits* the execution details, e.g., the specified well for washing.
- An ISA which exploits the parallel nature in the art of experiment.

1.1 AutoBio ISA Overview

The central part of the base AutoBio-ISA is the *execution graph* and *instruction*, we split the entire protocol into different semantic blocks, and translate the procedures written in natural language inside the block into AutoBio instructions. We exploit the block-level parallelism by checking the mutual interference between each block, creating the dependency graph, and finding the critical path on this graph, see chapter **??**.

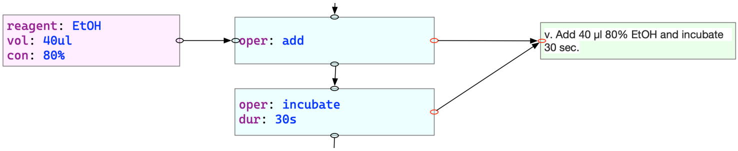

Abbildung 1.1: An example of instruction flow

1.2 Instruction Encoding

An *instruction* describes a specific form of a laboratory operation, including the operation itself (called a *"mnemonic"*), the input of the operation (called the *"slot"* of the action), and the output of the operation (called the *"emit"* of the action). As an example, the following instruction describes the procedure of reagent addition followed by incubation. The original form of the instruction flow is represented in a graph form where the edges denote the regent transferring from one step to another.

As shown in Abbildung 1.1, the only constraint between instructions is the flow of reagent. An instruction emits reagents, and another instruction catches them in its slots, forming a flow edge. To serialize the graph into a sequence, we assume that each instruction has exactly *one* or no output, and assign a unique number to each instruction. The sequential version of the instruction is

```
15: ADD %14, %5
16: INCUBATE %15
17: PLACE %16 %4 "freeze"
```

where the instruction `ADD` has the number `15` and its input slot catches the reagents emitted from the instruction `14` and `5`. After incubation, the resulting reagent is placed in the condition specified in 4. The optional field `"freeze"` comes from the protocol text, indicating the `PLACE` operation in the instruction list is originally described as "freeze".

The instruction has the form `MNEMONIC SLOTS LITERAL`, where the optional field `LITERAL` records the verb used in the original protocol, the `MNEMONIC` field is a pre-defined value selected from a set of mnemonic we defined in Chapter 3.

1.3 Reagent Encoding

TODO

2 Data Types

This chapter describes the data types defined for AutoBio-ISA. There are basic types and structural types.

2.1 Basic Types

2.1.1 Booleans

2.1.2 Numbers

2.1.3 Temperature Specifications

2.1.4 Pressure Specifications

2.1.5 Volume Specifications

2.1.6 Duration Specifications

2.2 Structural Types

2.2.1 Reagents

2.2.2 Conditions

3 Basic Actions

This chapter describes the basic instructions used in biological experiments. We aim to reduce concrete actions into abstract operations, without loss of their executability.

An instruction may have some slots (inputs) or emits (outputs), and each slot or emit has a specified data type attached to it.

The `REQ` flag denotes that this argument is *Required*, the `OPT` flag denote that this argument is *Optional*, and the `RET` flag denote that this argument is an emit (return value).

The `REG, COND, BOOL` flag represents the type of *Reagent, Condition* and *Boolean*, respectively.

Each instruction corresponds to an instruction node in the execution graph.

3.1 add

```
r3 = ADD r1, r2, Ordered=True
```

Description

Add a reagent to another reagent. This action is the basic abstraction of adding a reagent to another. Examples include mixing reagents, applying a drop, etc.

Slots & Emits

Argument	E/S	Type	Description
`r1`	Slot	`REQ|REG`	The *target* reagent.
`r2`	Slot	`REQ|REG`	The *additive* reagent.
`b3`	Slot	`OPT|BOOL`	`r1` and `r2` are interchangeable.
`r4`	Emit	`RET|REG`	The *result* reagent.

Examples

P/I	Content
Proto.	Apply a drop of normal goat serum to each specimen.
Instr.	`ADD [specimen], [goat serum], b3=True`[1]
Proto.	Load supernatant on to 3 ml equilibrated Ni-NTA column.
Instr.	`ADD [Ni-NTA], [supernatant], b3=True, LOAD`

[1]The `[specimen]` and `[goat serum]` are two reagent objects holding their solutes, volumes, etc..

3.2 concentrate

```
r3 = CONCENTRATE r1, c2
```

Description

Concentrate a reagent to a specified condition.

Slots & Emits

Argument	E/S	Type	Description
`r1`	Slot	`REQ|REG`	The reagent to be concentrated.
`r2`	Slot	`REQ|COND`	The target condition.
`r3`	Emit	`RET|REG`	The *result* reagent.

Examples

P/I	Content
Proto.	Adjust the cells concentration into 1.5×10^5 cells/mL.
Instr.	`CONCENTRATE [cells], [concentration=`1.5×10^5`]`
Proto.	Concentrate eluate in speedvac to 22.5 μL
Instr.	`CONCENTRATE [eluate], [volume=`$22.5\mu L$`]`

3.3 incubate

```
r3 = INCUBATE r1, c2
```

Description

Incubate a reagent in a specified environment.

Slots & Emits

Argument	E/S	Type	Description
`r1`	Slot	`REQ|REG`	The reagent to be incubated.
`r2`	Slot	`REQ|COND`	The incubation environment.
`r3`	Emit	`RET|REG`	The *result* reagent.

Examples

P/I	Content
Proto.	Incubate at 37°C for 2-3 minutes.
Instr.	`INCUBATE [cells], [temp=37C, dur=2-3min]`
Proto.	Incubate the mixture for 5 min at room temperature.
Instr.	`INCUBATE [cells], [temp=ROOM, dur=5min]`

3.4 elute

```
r4 = ELUTE r1, r2, c3
```

Description

Elute the reagent r1 in the eluent r2 under the condition c3, the resulting reagent is given in r4.

Slots & Emits

Argument	E/S	Type	Description
r1	Slot	REQ|REG	The reagent to be eluted.
r2	Slot	REQ|REG	The eluent reagent.
c3	Slot	OPT|COND	The environment condition for elution.
r3	Emit	RET|REG	The result eluate reagent.

Examples

P/I	Content
Proto.	Elute the histone complex with 250 μL freshly prepared elution buffer.
Instr.	ELUTE [complex], [elution buffer]
Proto.	Elute RNA in 20 μL of Elution Solution A.
Instr.	ELUTE [rna], [elution solution A]

3.5 wash

```
r4 = WASH r1, r2, c3
```

Description

Wash the reagent `r1` in the fluid reagent `r2` under the condition `c3`, the resulting reagent is given in `r4`.

Slots & Emits

Argument	E/S	Type	Description
`r1`	Slot	`REQ|REG`	The reagent to be washed.
`r2`	Slot	`REQ|REG`	The wash fluid.
`c3`	Slot	`OPT|COND`	The environment condition for washing.
`r3`	Emit	`RET|REG`	The washed reagent.

Examples

P/I	Content
Proto.	Wash the cell in 1ml PCR-grade water.
Instr.	`WASH [cell], [PCR-grade water]`
Proto.	Wash the digested tissues with HBSS containing 5 mM EDTA.
Instr.	`WASH [tissues], [HBSS]`

3.6 transfer

```
r3 = TRANSFER r1, r2
```

Description

Transfer the reagent `r1` to the reagent `r2`, the resulting reagent is given in `r3`.[2]

Slots & Emits

Argument	E/S	Type	Description
`r1`	Slot	`REQ|REG`	The reagent to be transferred.
`r2`	Slot	`REQ|REG`	The target container.
`r3`	Emit	`RET|REG`	The transferred reagent.

Examples

P/I	Content
Proto.	Transfer the aqueous phase to a tube containing 500μL isopropanol.
Instr.	`TRANSFER [aqueous phase], [tube]`
Proto.	Transfer the mixture to a phase-lock tube.
Instr.	`TRANSFER [mixture], [phase-lock tube]`

[2] The reagent `r2` might be an empty container.

3.7 spin

```
r3 = SPIN r1, c2
```

Description

Spin (or centrifugate) the reagent `r1` under the condition `c2`, the resulting reagent is given in `r3`.

Slots & Emits

Argument	E/S	Type	Description
`r1`	Slot	`REQ|REG`	The reagent to be centrifugated.
`c2`	Slot	`REQ|COND`	The centrifugation settings.
`r3`	Emit	`RET|REG`	The resulting reagent.

Examples

P/I	Content
Proto.	Centrifuge for 5 min 30 sec at 230 rcf to pellet cells.
Instr.	`SPIN [cells], [5m30s,230rcf]`
Proto.	Spin down the plates at 110 rcf for 6 min at room temperature.
Instr.	`SPIN [plate], [6m,110rcf,room temp]`

Important Notes

We treat *Spin* and *Centrifuge* as the same operation.

3.8 collect

```
r3 = COLLECT r1, r2
```

Description

Collect the reagent `r1` to the reagent (or container) `r2`, the resulting reagent is given in `r3`.

Slots & Emits

Argument	E/S	Type	Description
`r1`	Slot	`REQ|REG`	The reagent to be collected ('collect-from').
`r2`	Slot	`OPT|REG`	The target reagent or container ('collect-to').
`r3`	Emit	`RET|REG`	The resulting reagent.

Examples

P/I	Content
Proto.	Collect suspensions into 2 x 250 mL conical bottles.
Instr.	`COLLECT [suspension], [conical bottle]`
Proto.	Collect resulting emulsion via PE2 tubing into a 50 mL Falcon tube.
Instr.	`COLLECT [emulsion], [Falcon tube]`
Proto.	Collect livers from e13.5 mouse embryos.
Instr.	`COLLECT [e13.5 mouse embryos], [livers]`

Important Notes

Clarification between `COLLECT` and `TRANSFER` needs to be provided. There exists ambiguity in the above examples.

3.9 shake

```
r3 = SHAKE r1, c2
```

Description

Shake the reagent `r1` under the condition `c2`, the resulting reagent is given in `r3`.

Slots & Emits

Argument	E/S	Type	Description
`r1`	Slot	`REQ|REG`	The reagent to be shaken.
`c2`	Slot	`REQ|COND`	The shake settings.
`r3`	Emit	`RET|REG`	The resulting reagent.

Examples

P/I	Content
Proto.	Shake the lymph ducts gently.
Instr.	`SHAKE [lymph ducts], [gently]`
Proto.	Shake vigorously for 15 seconds.
Instr.	`SHAKE [cell], [vigorously,15sec]`

Important Notes

`SHAKE` might be both an action or a field of `COND` (with shaking).

3.10 suspend

```
r3 = SUSPEND r1, r2
```

Description

Suspend the reagent `r1` using the reagent `r2`, the resulting reagent is given in `r3`.

Slots & Emits

Argument	E/S	Type	Description
`r1`	Slot	`REQ|REG`	The reagent to be suspended.
`r2`	Slot	`REQ|REG`	The suspension liquid.
`r3`	Emit	`RET|REG`	The resulting reagent.

Examples

P/I	Content
Proto.	Aspirate supernatant and suspend cells in PGC medium.
Instr.	`SUSPEND [cells], [PGC medium]`
Proto.	Suspend frozen yeast cell pellets in 5 ml lysis solution.
Instr.	`SUSPEND [cells], [lysis solution]`

3.11 dialyze

```
r3 = DIALYZE r1, r2, c3
```

Description

Dialyze the reagent `r1` using the reagent `r2`, the condition for dialysis is given in `c3`.

Slots & Emits

Argument	E/S	Type	Description
`r1`	Slot	`REQ|REG`	The reagent to be dialyzed.
`r2`	Slot	`REQ|REG`	The dialysis solution.
`c3`	Slot	`OPT|COND`	The condition for dialysis.
`r4`	Emit	`RET|REG`	The result reagent.

Examples

P/I	Content
Proto.	Dialize flow through in FPLC Buffer A.
Instr.	`DIALYZE [flow], [FPLC Buffer A]`
Proto.	Dialyze protein into appropriate experimental buffer.
Instr.	`DIALYZE [protain], [appropriate buffer]`

4 Pseudo Actions

4.1 repeat

```
r3 = REPEAT s1, r2
```

Description

Repeat the

Slots & Emits

Argument	E/S	Type	Description
r1	Slot	REQ|REG	The reagent to be collected ('collect-from').
r2	Slot	OPT|REG	The target reagent or container ('collect-to').
r3	Emit	RET|REG	The resulting reagent.

Examples

Important Notes

Clarification between `COLLECT` and `TRANSFER` needs to be provided.

There exists ambiguity in the above examples.

P/I	Content
Proto.	Collect suspensions into 2 x 250 mL conical bottles.
Instr.	`COLLECT [suspension], [conical bottle]`
Proto.	Collect resulting emulsion via PE2 tubing into a 50 mL Falcon tube.
Instr.	`COLLECT [emulsion], [Falcon tube]`
Proto.	Collect livers from e13.5 mouse embryos.
Instr.	`COLLECT [e13.5 mouse embryos], [livers]`

5 Analytical Actions

measure

analyse

plot

fit