

SBML Level 2 Array Features Proposal

Andrew Finney^{1,2}, Victoria Gor³, Ben Bornstein³
Eric Mjolsness³, Hamid Bolouri^{1,2}

¹ California Institute of Technology

² University of Hertfordshire

³ Jet Propulsion Laboratory

Motivation

- Model the logical connectivity in tissues and simple organisms where the detailed geometry is not significant.
 - For example:
 - the community effect in developmental gene regulation
 - (Gurdon 1988)
- Enable the creation of abstract models

Overview

This proposal includes the following features:

- **Arrays of species, compartments, reactions and parameters**
 - And instances when combined with Modularity proposal
- **Attribute values varying across values in an array**
- **Sparse Arrays**
- **Connectivity described as a sparse array**
- **Syntax Sugar simplifies models**
- **Rules applied to regions of arrays**
 - Not covered here
- **New Built-in functions**
 - `sum`, `product`, `reduce`, `map`, `argmin`, `argmax`
 - Not covered here

Domain Definitions

Model
name : SName { use= "optional" } domain : Domain[0..*] ...

New Elements:

Domain
symbol : Symbol[1..*] lowerBound : Integer upperBound : Integer

Symbol
symbol : SName

Domain Symbols

- These are symbols declared in domain elements
- Domain symbols share the same namespace as all other symbols
- Values of domain symbols don't vary during a simulation - they are considered constant
- A constant expression is one which contains only constant symbols

Domain Definition Example

```
<model name="tissue">
  <listOfDomains>
    <domain upperBound="10" lowerBound="-10">
      <listOfSymbols>
        <symbol name="x"/>
        <symbol name="y"/>
      </listOfSymbols>
    </domain>
  </listOfDomains>
  ...
</model>
```

Array Declaration

- **New Array operator [] used with symbol in name field**
 - similar to C
 - Number of operators indicates the number of dimensions
 - Domain symbol inside [] defines size of array
 - Attributes and the attributes of sub-elements are the scope of the domain symbol
- **Can be applied to species, compartments, reactions and parameters**
- **Rules are a special case – not covered here**

Array Declaration - Example

```
<model name="simple">
  <listOfDomains>
    <domain upperBound="9" lowerBound="0">
      <listOfSymbols>
        <symbol name="x"/>
      </listOfSymbols>
    </domain>
  </listOfDomains>
  <listOfCompartments>
    <compartment name="cell[x]"/>
  </listOfCompartments>
  ...
</model>
```

Symbol Scope - Example

```
<model name="ref">
  <listOfDomains>
    <domain upperBound="9" lowerBound="0">
      <listOfSymbols>
        <symbol name="x"/>
      </listOfSymbols>
    </domain>
  </listOfDomains>
  <listOfCompartments>
    <compartment name="cell[x]" volume="x * 0.2" />
  </listOfCompartments>
  ...
</model>
```

Simple Sparse Arrays

```
<model name="notsimple">
  <listOfDomains>
    <domain upperBound="9" lowerBound="0">
      <listOfSymbols>
        <symbol name="x"/>
        <symbol name="y"/>
      </listOfSymbols>
    </domain>
  </listOfDomains>
  <listOfCompartments>
    <compartment name="cell2D[x][y]"/>
    <compartment name="cellDiagonal[x][x]"/>
  </listOfCompartments>
  ...
</model>
```

Referring to Array Elements

- We can refer to specific elements of an array in all the places where symbol names can be used to refer to an object
 - For Example
 - Specie attribute value on specieReference
 - Formula attribute value on kineticLaw
- Use the Array operator [] again
- A numeric expression is used inside the operator to indicate a specific element of the array

Referring to Array Elements Example

```
<model name="ref">
  <listOfDomains>
    <domain upperBound="9" lowerBound="0">
      <listOfSymbols>
        <symbol name="x"/>
      </listOfSymbols>
    </domain>
  </listOfDomains>
  <listOfCompartments>
    <compartment name="cell[x]"/>
  </listOfCompartments>
  <listOfSpecies>
    <species name="s[x]" compartment="cell[x]"/>
  </listOfSpecies>
  ...
</model>
```

Conditional Sparse Arrays

- Basis for connection scheme
- All elements that can declare an array have additional attribute `elementExists`.
- `elementExists` contains constant expression
 - Boolean value indicates whether element exists
 - perhaps in later versions of SBML expression can be dynamic?

Conditional Sparse Array Example

```
<model name="starfish">
  <listOfDomains>
    <domain upperBound="9" lowerBound="0"/>
      <listOfSymbols>
        <symbol name="x"/>
        <symbol name="y"/>
      </listOfSymbols>
    </domain>
  </listOfDomains>
  <compartment name="cell[x][y]"
    elementExists="y <= x">
    ...
  </model>
```

Example of using Conditional Sparse Array to represent connections

```
<model name="tissue">
  <listOfDomains>
    <domain upperBound="9" lowerBound="0">
      <listOfSymbols>
        <symbol name="x" />
        <symbol name="y" />
        <symbol name="x1" />
        <symbol name="x2" />
        <symbol name="y1" />
        <symbol name="y2" />
      </listOfSymbols>
    </domain>
  </listOfDomains>
  <listOfCompartments>
    <compartment name="grid[x][y]" />
  </listOfCompartments>
</model>
```

Example continued

```
<listOfSpecies>  
  <species  
    name="s[x][y]"  
    initialAmount="0.1"  
    compartment="grid[x][y]"/>  
</listOfSpecies>
```

Example continued

```
<listOfReactions>
  <reaction
    name="connections [x1] [y1] [x2] [y2]"
    elementExists=
"abs (x2 - x1) == 1 || abs (y2 - y1) == 1">

    <listOfReactants>
      <specieReference specie="s [x1] [y1]" />
    </listOfReactants>
    <listOfProducts>
      <specieReference specie="s [x2] [y2]" />
    </listOfProducts>
    <kineticLaw formula="s [x1] [y1] * 0.1" />

  </reaction>
</listOfReactions>
</model>
```

Implied Arrays

- Syntax Sugar
- Can be applied to species, compartment and parameters
- Array form is 'implied' by the structure containing a given object
- Loss of flexibility

Implied Species Arrays - Example

Given

```
<compartment name="cell[x][y]" />
```

the structure

```
<specie name="s[x][y]"  
  compartment="cell[x][y]"  
  initialAmount="x == 0 ? 1e-10 : 0" />
```

can be replaced with the equivalent structure

```
<specie name="s"  
  compartment="cell"  
  initialAmount="x == 0 ? 1e-10 : 0" />
```

Referencing Nested Array Elements

- More Syntax sugar
- `'.'` operator
 - Left hand side is array element or object
 - Right hand side is object contained inside
 - More readable
 - Issue: compatibility with modularity proposal

Dot operator - Example

Given

```
<listOfCompartments>  
  <compartment name="cell[i]"/>  
</listOfCompartments>  
<listOfSpecies>  
  <species name="s1" compartment="cell"/>  
</listOfSpecies>
```

then

```
<specieReference specie="s1[i]"  
  stoichiometry="1"/>
```

can be replaced by

```
<specieReference specie="cell[i].s1"  
  stoichiometry="1"/>
```

Summary of Proposal

This proposal includes the following features:

- Arrays of species, compartments, reactions and parameters
 - And instances when combined with Modularity proposal
- Attribute values varying across values in an array
- Sparse Arrays
- Connectivity described as a sparse array
- Syntax Sugar simplifies models
- Rules applied to regions of arrays
 - Not covered here
- New Built-in functions
 - `sum`, `product`, `reduce`, `map`, `argmin`, `argmax`
 - Not covered here