Modelando Unidades de Processamento Local no cérebro da Drosófila Melanogaster em CircuitML

Daniel Salles Chevitarese
IBM Research Brazil
Rio de Janeiro, Brasil
daniel@chevitarese.com.br

Marley Vellasco - IEEE Member
Departamento de Engenharia Elétrica - PUC-Rio
Rio de Janeiro, Brasil
marley@ele.puc-rio.br

***Resumo*—O mapeamento dos diversos circuitos neurais em unidades funcionais tem permitido cientistas estudarem esse sistema tão complexo. Por apresentar um repertório comportamental rico desempenhado por um circuito relativamente pequeno, a Drosófila Melanogaster se mostra como um estudo de caso interessante. Análises do conectoma da mosca, utilizando ferramentas de marcação genética e ferramentas avançadas de registro eletrofisiológico, permitiram que o circuito neural da mosca fosse mapeado em unidades funcionais, chamadas de Unidades de Processamento Local (LPU). Existem várias ferramentas disponíveis que permitem neurocientistas criarem um modelo preciso de todo o cérebro da mosca, mas nenhum deles fornece um método para especificar os circuitos de uma forma que ambos, biólogos e engenheiros, possam trabalhar juntos. Além disso, o desenvolvimento de modelos de LPU plausíveis requer a habilidade de especificar e instanciar sub-circuitos sem referência explícita a seus neurônios constituintes e suas ligações internas. Para este fim, apresentamos uma linguagem de especificação de circuito neural, chamada CircuitML, para construção de LPUs. A CircuitML foi concebida como uma extensão para NeuroML; a linguagem CircuitML proporciona construtos para definir sub-circuitos que compreendem primitivas neurais suportadas pela linguagem NeuroML. As LPUs e os sub-circuitos possuem uma interface que permite conexões com outras unidades funcionais através de padrões de conectividade definidos por blocos de conexão. A CircuitML foi usada para especificar um modelo modular baseado no sistema olfativo da mosca.**

Palavras-chave*—*Drosófila Melanogaster, Modelos, Simulação, XML, Python.

I. Introdução

A pesquisa sobre a mosca Drosófila transformou este inseto minúsculo em um modelo genético muito bem compreendido e manipulável [1], o que revela muitos princípios de desenvolvimento, regulação genética e sinalização celular. Tais princípios são conservados entre espécies [1] e podem ajudar os cientistas a entender cérebros mais complexos e explicar algumas doenças hereditárias.

Além desse poderoso ferramental para manipulação genética, avanços recentes em métodos experimentais para gravações precisas de respostas a estímulos neuronais da mosca [2], [3], [4], [5], bem como avanços em técnicas de análise de respostas comportamentais da mosca a estímulos [6], [7], [8], [9] vêm facilitando a identificação dos circuitos cerebrais. Além disso, o progresso na reconstrução do conectoma da mosca [10], [11], usando neurônios estereotipados, aqueles que são encontrados em todas as moscas [12], tem contribuído muito para a modelagem do circuito.

Do lado da engenharia, o equilíbrio entre complexidade e riqueza comportamental também valoriza o uso das moscas, uma vez que elas apresentam comportamentos facilmente observados desempenhados por, aproximadamente, 150 mil neurônios [13], o que significa cinco ordens de grandeza a menos que os vertebrados.

Estudos sobre o cérebro da Drosófila Melanogaster revelaram que este compreende cerca de 40 módulos funcionais distintos, a maioria dos quais correspondem a regiões anatômicas no cérebro que são associadas com sistemas sensoriais específicos, além de locomoção. Estes módulos são referidos como Unidades de Processamento Local (LPUs), porque eles possuem uma população bem característica de neurônios locais. Dado que muitas LPUs são associadas com comportamentos específicos, elas podem ser consideradas os blocos de construção do cérebro da mosca. Além disso, percebe-se que muitos desses circuitos são organizados em sub-circuitos que se repetem e integram as funções de cada módulo. Sendo assim, para modelar essas LPUs, é necessário uma linguagem que permita a especificação e conexão de múltiplas instâncias de sub-circuitos sem a necessidade de se recriar todo o circuito dentro do mesmo.

Uma importante ferramenta que está disponível para a especificação de sistemas nervosos é a NeuroML [14], uma meta-linguagem, baseada em XML (Extensible Markup Language). Embora muito simples e fácil de usar, a linguagem é muito poderosa, porque permite que redes neurais e seus componentes sejam definidos de forma detalhada para serem usados em vários simuladores (NEURON [15], GENESIS[16], MOOSE [17], NEST [18]), através de arquivos de especificação padronizados [14]. Essa padronização resolve diversos problemas de compatibilidade entre ferramentas de software, facilitando a reprodução dos modelos já publicados em NeuroML e seus resultados. Além disso, essa característica permite o compartilhamento e reutilização de modelos já especificados, facilitando o desenvolvimento de novos modelos [14].

Embora a NeuroML aborde o problema de compatibilidade de muitas maneiras, ela tem suas limitações quanto à especificação das unidades de processamento local, uma vez que se destina a tratar dos diversos sistemas neurológicos que são organizados de forma biológica. Para se especificar as unidades funcionais, abstraindo seus sub-circuitos canônicos,

é necessária uma ferramenta que ofereça tanto uma linguagem padronizada quanto um suporte para a criação de componentes novos que atendam ao ponto de vista da engenharia. Este é o principal objetivo da nova linguagem de especificação de circuitos neurais, chamada CircuitML, apresentada neste artigo.

II. Criando as unidades funcionais

A linguagem CircuitML foi desenvolvida juntamente com um arcabouço para a modelagem de cérebros virtuais como um conjunto de blocos funcionais, em vez de representá-los como redes de neurônios interconectados. Cada bloco de construção compreende componentes menores, permitindo que se possa estudar, por exemplo, situações no formato "IF-THEN", como proposto por [1]: "se este neurônio é removido, que comportamento será afetado?" ou "se alguém altera o sistema de detecção de movimento, então...", ou mesmo "se mais canais são acrescentados [4] ao sistema olfativo da mosca, então...".

Para atingir este novo nível de abstração, a CircuitML foi desenvolvida como uma linguagem de descrição estruturada, que teve seus conceitos iniciais publicados em [19]. A linguagem pode descrever circuitos neurais em um nível acima do NeuroML nível 3 (NetworkML), herdando o seu apoio a muitas ferramentas e simuladores, e permitindo aos cientistas compartilhar e validar suas descobertas [14]. Além disso, tal herança dá a CircuitML uma grande variedade de elementos, variando de morfologias neuronais, com MorphML [14], para um cérebro inteiro compreendido por LPUs.

Como uma ferramenta complementar para CircuitML, também foi desenvolvida uma API Python, chamada libCircuitML. O principal objetivo do libCircuitML é fornecer facilidade de uso de utilitários para a manipulação de especificações em CircuitML, usando ferramentas em Python, familiares para os programadores. Essa ferramenta pode ser importada para um script Python que permite que os usuários:

- carreguem e validem arquivos CircuitML;
- editem especificações e analisá-las através do esquema CircuitML;
- salvem arquivos CircuitML válidos tanto para o esquema CircuitML quanto para o esquema NeuroML;
- usem funcionalidades adicionais através de scripts Python;
- conectem ao Neurokernel – um simulador voltado para validação de modelos do cérebro da Drosófila [19], [20] para simulação.

A. Visão geral da CircuitML

O documento CircuitML consiste em elementos XML descrevendo cada um dos componentes do sistema neural. As estruturas desses componentes são definidas a partir de um esquema XML (XSD), que é manipulado por bibliotecas já existentes para verificar a validade dos documentos. Por exemplo, se algum nome de atributo obrigatório estiver faltando em qualquer elemento definido, o validador informa o erro e onde o mesmo ocorreu.

Uma vez que o documento XML está de acordo com o formato definido pelo esquema CircuitML, o arquivo pode ser transformado para outros formatos de várias maneiras, entre elas a SAX (Simple API for XML) ou a DOM (Document Object Model). Sendo assim, as especificações em CircuitML podem ser portadas para diversos simuladores, incluindo NEURON, GENESIS e PSICS. Essa abordagem permite que diversas aplicações suportem a linguagem sem que seja preciso alterar o código fonte dessas aplicações.

Antes de descrever a CircuitML, é importante entender as diferenças entre NeuroML e CircuitML. Ao encapsular todo o sistema nervoso em LPUs, a CircuitML auxilia cientistas a enxergar o sistema como um grande circuito construído a partir de chips (LPUs). Cada "chip" provê sua própria funcionalidade através de interfaces bem definidas. Essa característica da linguagem faz referência ao início do estudo de orientação a objetos, quando Booch [21] afirmou que o paradigma em questão ajudaria a gerenciar a complexidade de sistemas massivamente grandes. Além disso, enquanto o NeuroML fornece todas as declarações das primitivas e suas respectivas funcionalidades, a CircuitML provê toda a abstração necessária para tornar o processo de especificação mais simples. As vantagens da CircuitML sobre o NeuroML são:

- minimização do acoplamento entre elementos de circuito ("chips");
- interfaces claramente definidas, permitindo a abstração de detalhes dos circuitos dentro de elementos maiores;
- reuso e limpeza do código.

O escopo do CirctuiML cobre a definição de módulos funcionais (LPUs) com interfaces, módulos menores, chamados sub-circuitos, e um módulo de conexão que conecta dois módulos. A Fig. 1 mostra a estrutura geral da CircuitML, que será explicada na próxima seção.

B. Elementos CircuitML

A CircuitML implementa quatro elementos principais. Tais elementos estão dispostos na Fig. 1:

1) ***lpu*** (caixa preta): encapsula a unidade funcional com uma interface que expõe neurônios de entrada e saída;
2) ***subcircuit*** (caixa cinza): encapsula partes menores, chamadas sub-circuitos, para serem reutilizados dentro de LPUs. Sub-circuitos podem conter outros sub-circuitos, populações de neurônios, sinapses e outros elementos NeuroML;
3) ***interface*** (caixa cinza): define um relacionamento entre as portas de entrada e saída, e os elementos públicos internos da LPU;
4) ***connectivity*** (caixa branca): descreve o mapa de conexões entre duas LPUs.

1) Elemento LPU: A Fig. 2 apresenta um exemplo de uso do elemento ***lpu*** para especificar um módulo chamado *detector de companheiro* contendo 3 blocos principais: (1) **TEM** (Time Encoding Machine), usado para codificar o sinal analógico da entrada em pulsos, (2) **Classificador** que processa a saída do TEM e classifica como *parceiro* ou *não parceiro* e (3) **pré-motor** que processa a saída do Classificador e envia para portas de saída da LPU. Nesse exemplo, o Classificador julga

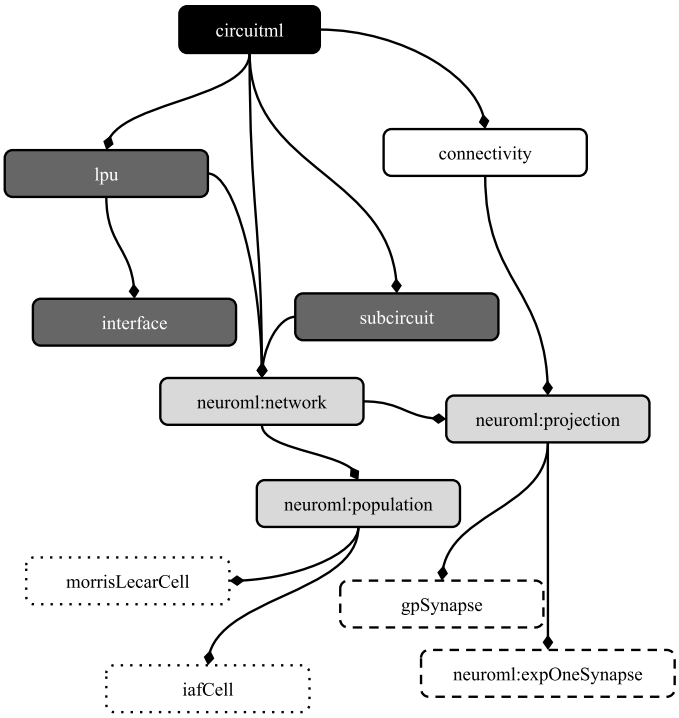

Fig. 1. Visão geral da estrutura CircuitML. O elemento no nível superior, *circuitml* (preto), contém elementos de vários tipos. Populações (*populations*) encapsulam células e projeções (*projections*) (cinza claro), sinapses. Populações são encapsuladas por sub-circuitos (*subcircuit*) ou LPUs (*lpu*) (cinza). Na caixa branca, o elemento de conexão *connectivity* encapsula projeções.

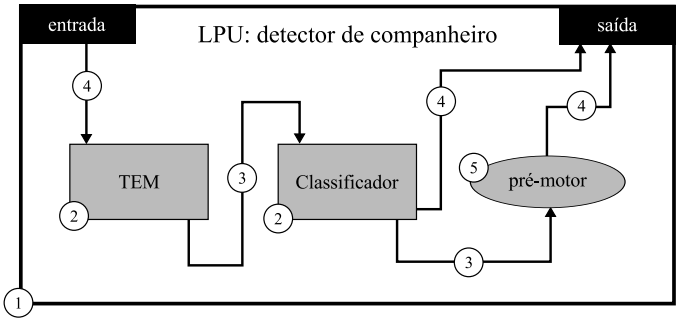

Fig. 2. Exemplo de um LPU fictícia (1) que detecta se a entrada corresponde ao tipo esperado. As caixas (2) compreendem circuitos com algumas função associada e a elipse cinza representa uma rede neural que envia informações para a saída da LPU. As flechas numeradas com 4, referem-se a projeções que entram ou saem de portas das interfaces, flechas numeradas com 3 referem-se a projeções entre sub-circuitos, e entre populações de células. A elipse numerada com 5 representa uma população de neurônios.

como *parceiro* a entrada gerada por um animal da mesma espécie e como *não parceiro* o contrário. Um sistema similar pode ser encontrado na mosca Drosófila [22], [23], [24], [25], [26], [27].

2) Elemento de sub-circuito: Além de redes neurais, LPUs podem conter pequenas partes que se repetem e que são chamadas de sub-circuitos. O elemento ***subcircuit*** define esse tipo de estrutura. Eles são muito similares ao elemento que define redes (*neuroml:network*), mas podem abrigar elementos personalizados que não necessariamente foram construídos com as primitivas do NeuroML, como, por exemplo, o

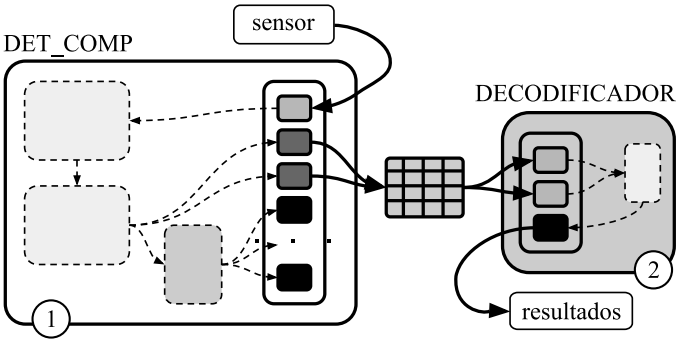

Fig. 3. Duas LPUs interconectadas por um elemento ***connectivity***: a LPU 1 refere-se a um detector de parceiro e a LPU 2, a um decodificador. Circuitos internos são mostrados como caixas tracejadas, porque eles não são visíveis do lado de fora das LPUs.

codificador TEM e o Classificador, ambos na Fig. 2. Essa habilidade da linguagem CircuitML é importante ao se especificar o sistema sensorial, uma vez que a entrada precisa ser transformada em pulsos ou em um tensão a ser apresentada na membrana das células.

Note que os elementos *subcircuit* não são LPUs. Sub-circuitos não possuem interfaces e expõem todo o seu circuito interno, o que acaba com a abstração. Além disso, eles podem ser usados apenas para reuso e não possuem uma funcionalidade bem definida, o que contraria o ponto principal de uma LPU.

3) Elemento de interface: O elemento ***interface*** pode ser entendido como um mapa entre o circuito externo e as partes internas do chip que a possui. A Lista 1 apresenta um exemplo de uma interface expondo 4 elementos internos através de, respectivamente, 2 portas de entrada e 2 portas de saída. Na linha 2, a porta `ent_0` expõe o primeiro neurônio da população `minha_pop`, e na linha 4, a `sai_2` expõe a saída do quarto neurônio. Note que, nesse exemplo, o terceiro neurônio está inacessível de fora da LPU por não estar listado na interface `minha_interface`.

Lista 1. Exemplo de uma interface

```
1 <interface id="minha_interface">
2   <port id="ent_0" in="0" out="minha_pop/0"/>
3   <port id="ent_1" in="1" out="minha_pop/1"/>
4   <port id="sai_0" in="minha_pop/3" out="2"/>
5   <port id="sai_1" in="minha_pop/4" out="3"/>
6 </interface>
```

4) Elemento de conectividade: O último elemento apresentado aqui é o ***connectivity***, que pode ser entendido como o bus que conecta dois chips. Ele é responsável por encapsular *projections* e *connections* que não podem ser definidos fora das LPUs e conectá-los a portas de entrada ou saída das LPUs. Na Fig. 3, a LPU *detector de companheiro* recebe sinais analógicos de um sensor externo (Fig. 3 - caixa superior) e envia a classificação *parceiro/não parceiro* para outra LPU que processa essa informação e apresenta de alguma forma os resultados.

Na Lista 2, o elemento *connectivity* indica como conectar as LPUs `det_comp` e `decodificador` - linhas 6 a 14. Isso significa que, para cada conexão, são definidos o tipo da sinapse e qual portas ela ligará.

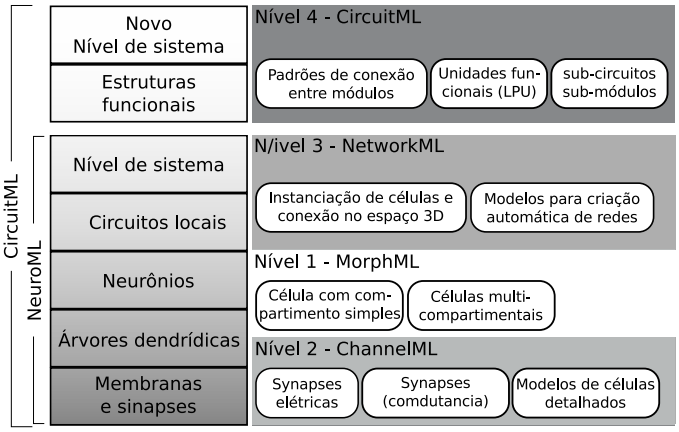

Fig. 4. Níveis de abstração em CircuitML e suas relações, na escala biológica, com o sistema nervoso.

Lista 2. Exemplo de duas LPUs interconectadas (Fig. 3)

```
<circuitml id="lpus">
 <Include href="det_comp.xml" />
 <Include href="decodificador.xml" />
 <expOneSynapse id="sinapse" erev="20mV"
     gbase="65nS" tauDecay="3ms" />
 <connectivity id="dc_decodificadoro"
     lpu1="det_comp" lpu2="decodificador">
  <connection from="det_comp/valido_out"
         to="decodificador/valido"
         synapse="sinapse"/>
  <connection from="det_comp/invalido_out"
         to="decodificador/invalido"
         synapse="sinapse"/>
 </connectivity>
</circuitml>
```

C. Níveis de abstração

Os novos componentes da CirctuiML permitem a abstração das unidades funcionais que escondem os circuitos internos e expõem neurônios e projeções para receberem ou enviarem informações através de portas. Sub-circuitos já definidos, contendo populações de neurônios interconectados, podem ser reutilizados dentro das LPUs, o que simplifica a especificação dos módulos. A Fig. 4 apresenta a relação entre os elementos da escala biológica e os elementos em CircuitML. A partir do menor nível biológico, são definidos os níveis ChannelMl (nível 2) e MorphML (nível 1), seguidos do maior nível da escala biológica, o NetworkML (nível 3); cada um desses níveis são definidos na pilha de abstração do NeuroML. Nesse trabalho é incluído um quarto nível, descrito pela CircuitML, para encapsular os níveis inferiores em unidades funcionais.

Para melhor entender como essa simplificação funciona, a Fig. 5 mostra um exemplo de abstração no sistema olfativo da mosca. Nessa figura, alguns elementos em CircuitML são usados para se recriar parte do sistema olfativo da mosca Drosófila; o lóbulo antenal (AL), que pode ser entendido como um elemento *lpu*. O AL é formado por canais, especificados como elementos do tipo *subcircuit*, que, por sua vez, compreendem grupos de neurônios - *population* [14]. Atrás da caixa com a LPU do lóbulo antenal, aparece parte do cérebro da Drosófila dividido em neurópilos delineados em preto e que, na maioria dos casos, funciona como uma unidade funcional [13]. Dentro da caixa, um modelo simples

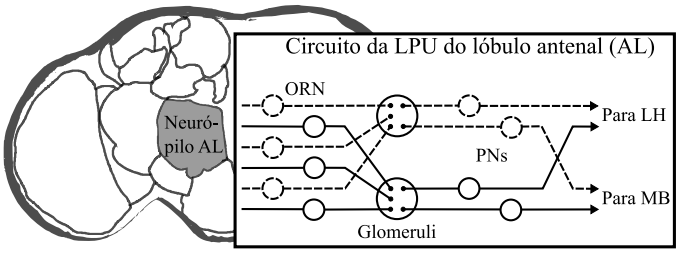

Fig. 5. Cérebro da mosca Drosófila com o vários neurópilos delineados e o lóbulo antenal esquerdo preenchido em cinza.

do lóbulo antenal esquerdo da mosca (*AL*) com dois canais (sub-circuitos). Cada canal, possui receptores olfativos (ORNs) e neurônios de projeção (PNs).

O quarto nível da pilha apresentada na Fig. 4 define os novos componentes que permitem a especificação dos módulos e suas interconexões. Esse nível também funciona como extensão do NeuroML, fornecendo mecanismos para encapsular redes neurais, definidas em NeuroML, em unidades funcionais com interfaces bem definidas. Nesse nível, áreas do cérebro são especificadas como LPUs que possuem uma funcionalidade em particular.

Como mencionado anteriormente, as unidades de processamento local podem ser comparadas a chips em um circuito. Cada chip possui sua funcionalidade implementada internamente, que é independente do circuito externo, e que é disponibilizada através de uma interface bem definida e documentada. Esse chips podem ter funcionalidades bem simples, mas com um número pequeno de diversos tipos dessa unidades combinadas, é possível criar sistemas complexos com uma enorme variedade de funções.

III. Resultados

Conforme descrito na seção anterior, os dois principais motivadores por trás da CircuitML são: modularidade funcional e conectividade. Assim, esta seção apresenta como essa modularização facilitou o desenvolvimento do sistema olfativo da mosca Drosófila.

Para ilustrar como as unidades funcionais funcionam, foi especificado o sistema olfativo primário da mosca Drosófila. O sistema em questão tem sido estudado por vários grupos [28], [29], [30], [31], mas o modelo apresentado é baseado no trabalho do grupo Bionet (Columbia University) publicado em [20].

O sistema olfativo primário da Drosófila contém dois lóbulos antenais, um em cada parte do cérebro, como mostrado na Fig. 5 como um neurópilo cinza. Cada uma dessas LPUs tem, aproximadamente, 49 glomérulos definidos morfologicamente que recebem informação de algo em torno de 30 receptores olfativos (ORN) que expressam sensibilidade ao mesmo odorante. Os axônios de cada ORN se conectam a árvores dendríticas de 3 a 5 neurônios PN dentro de um glomérulo que são responsáveis por transmitir a informação para outras regiões do cérebro: *mushroom body* (MB) e *lateral horn* (LH). Os ORNs também enviam informação para neurônios locais (LN) cujas conexões se restringem a outros neurônios dentro do lóbulo. O sistema olfativo primário da Drosófila contém cerca de 4.000 neurônios no total [31].

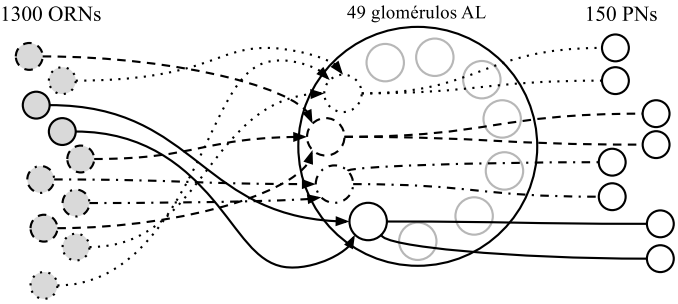

Fig. 6. Circuito do sistema olfativo de uma mosca adulta. Esse sistema é caracterizado pela convergência e divergência das conexões a partir do lóbulo antenal. Nessa figura, cada tipo de linha (sólida, pontilhada, tracejada, mista) é encapsulada em uma estrutura chamada canal [31] e representa grupos de neurônios que expressam o mesmo tipo de receptor, i.e., são ativados pelo mesmo tipo de molécula.

O modelo em CircuitML atual do sistema olfativo anterior da Drosófila especifica dois lóbulos antenais (AL e al) contendo 49 glomérulos cada - Lista 3. Por sua vez, cada canal encapsula, aproximadamente, 30 ORNs que enviam informação para 3 neurônios de projeção PNs, que podem ser vistos como a saída da atual especificação. O modelo inteiro compreende 2.800 neurônios, ou 70% do sistema real da mosca. Todos os neurônios foram descritos com o modelo Leaky Integrate-and-Fire (LIF) [32] e todas as correntes sinápticas foram modeladas usando-se funções-α [33] e conexões ponto-a-ponto. Os parâmetros de 24 canais foram baseados em dados disponíveis de ORNs [34]; e os outros parâmetros foram configurados artificialmente pelo grupo Bionet.

Na Fig. 6, cada canal é descrito por linhas sólidas, pontilhadas, tracejadas e mistas. Como discutido anteriormente, os lóbulos antenais contêm LN que se conectam a ORNs e PNs, e conexões entre os diversos tipos de neurônios (LN, ORN, PN). Ao se encapsular LNs, ORNs e PNs em canais, torna-se mais fácil perceber mecanismos pelos quais diferentes tipos de configurações de ORNs transformam percepção de odores em pulsos no cérebro da mosca, provocando um comportamento resultante. Eventualmente, ao se fornecer uma boa especificação do lóbulo antenal, tornar-se-á possível a abstração de tal módulo e a mudança do foco para centros secundários de processamento onde a entrada será a saída do módulo atual.

Na Lista 3, algumas linhas foram removidas para facilitar o entendimento. Na linha 3, a especificação referencia a especificação do canal, onde os ORNs, neurônios locais e PNs são descritos. Da linha 5 à linha 26, o `lobulo_esquerdo` é especificado com 1.300 entradas e 150 saídas, onde o padrão de conectividade de entrada e saída (E/S) e os neurônios internos da LPU são definidos da linha 8 à linha 19 através o elemento `es`. Considerando que todos os neurônios de projeção contidos em cada glomérulo podem emitir pulsos para outras LPUs, cada conexão dessa deve ser descrita dentro de `es`, onde cada porta será exposta externamente pela LPU. Neurônios locais ou ORNs não emitem sinais para fora das LPUs e, por isso, não devem ter portas associadas na interface `es`.

Lista 3. Versão simplificada do sistema olfativo.

```
1  <circuitml id="demo_olfato">
2   <!-- Incluindo especificacao externa -->
3   <Include href="demo_canal.xml" />
4   <!-- Primeira LPU: lobulo esquerdo -->
5   <lpu id="lobulo_esquerdo" input="1300"
6       output="150">
7    <!-- LPU E/S -->
8    <interface id="es">
9     <!-- saida -->
10    <port id="Or2a_c0" in="0" out="canal/0/ORN/0"/>
11    <!-- ... -->
12    <port id="Or2a_c49" in="1299"
13              out="canal/49/ORN/0"/>
14    <!-- saida -->
15    <port id="gl_0_0" in="canal/0/GL/0" out="0"/>
16    <!-- ... -->
17    <port id="gl_48_2" in="canal/48/GL/2"
18              out="149"/>
19   </interface>
20   <!-- Especificacao da rede -->
21   <network id="antena">
22    <!-- Instanciacao dos canais -->
23    <population id="canais" structure_type="canal"
24              size="49"/>
25   </network>
26  </lpu>
27  <!-- Segunda LPU: lobulo direito -->
28  <lpu id="lobulo_direito" input="1300"
29      output="150"><!-- IGUAL ANTERIOR -->
30  </lpu>
31 </circuitml>
```

Antes da CircuitML, um caminho seguido por muitos cientistas para se especificar sistemas em formatos padrão era a utilização de grafos, onde os nós são neurônios e os vértices, sinapses. O uso de grafos para representar esses sistemas apresenta algumas vantagens do ponto de vista da engenharia, mas não para quem especifica um sistema com milhares ou milhões de nós e vértices. Além disso, especificações com grafos, gastam muito mais espaço; por exemplo, usando o formato *gexf* (*http://gexf.net*), que é o formato padrão usado na Bionet, gasta-se 21.559 linhas para se especificar células, muitas repetidas uma vez que não se tem reuso, e 49.500 linhas para se especificar as sinapses.

Além da diferença do número de linhas gastas pela especificação do sistema, a CircuitML foi projetada para tornar a especificação muito mais clara. Na Lista 3 é fácil ter uma visão geral do sistema como um todo, i.e., LPUs com entrada e saídas claras e como os circuitos internos estão organizados interconectados. Outro fato percebido é que arquivos em CircuitML ocupam menos memória; no caso do sistema apresentado nesse artigo, a versão *gexf* ocupa 3MB não comprimida e 210KB comprimida, enquanto que a versão em CircuitML ocupa apenas 790KB não comprimida e 171KB comprimida. Percebe-se que a diferença entre as versões não compactadas é 389% menor, mas as compactadas é de apenas 20%. A razão para isso é que a especificação em CircuitML já reduz muito os dados através do reuso implícito na linguagem, e repetições no texto é muito explorado pelo compactadores.

IV. Conclusões e trabalhos futuros

Neste artigo foi apresentada a linguagem para a especificação de circuitos neurais CircuitML, que estende a NeuroML com o objetivo de oferecer modelos funcionais do cérebro que funcionam como blocos de construção. Cada bloco, apresentado como uma unidade de processamento local (LPU), abriga elementos que variam de simples populações de células a circuitos complexos com múltiplas camadas

de abstração. Embora as LPUs possam ser bem complexas internamente, suas interfaces escondem tal fato, expondo apenas as partes necessárias para o correto funcionamento das unidades. A partir das LPU, suas interfaces são interconectadas através de padrões que podem ser reusados, o que faz da CircuitML um linguagem bem poderosa para a especificação dos circuitos neurais.

Dadas as razões apresentadas nesse artigo, a Drosófila Melanogaster se mostra um bom modelo de estudo de caso da CircuitML. A especificação do sistema sensorial inteiro da mosca, com suas peculiaridades e interconexões, deve mostrar o quanto a CircuitML consegue abranger dessas especificações e o que falta se desenvolvido. Além disso, considerando que muitas dessas partes ainda estão faltando, a CircuitML pode ajudar cientistas a reconstruir o cérebro da mosca oferecendo suporte aos diversos blocos funcionais que forem surgindo, testando e validando tais módulos de forma independente. Outro ponto positivo é o compartilhamento das descobertas entre grupos de pesquisa na forma de blocos funcionais, o que permite que diferentes grupos abstraiam o conteúdo interno de LPU importadas e as utilize através de suas interfaces bem definidas.

Finalmente, o exemplo apresentado aqui, que descreve o sistema olfativo primário, já mostra que especificar sistemas neurais como circuitos facilita a modelagem e torna o sistema mais simples de ser compreendido. Como alguns grupos estudam o sistema visual da mosca Drosófila, o próximo passo será desenvolver a especificação do sistema visual e a integração deste com o sistema olfativo apresentado.

REFERÊNCIAS

[1] J. D. Armstrong, J. I. van Hemert, J. D. Armstrong, and J. I. van Hemert, "Towards a virtual fly brain," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 367, no. 1896, pp. 2387–2397, 2009.

[2] A. J. Kim, A. A. Lazar, and B. S. Yevgeniy, "2d encoding of concentration and concentration gradient in drosophila orns," in *Computational and Systems Neuroscience Meeting, Salt Lake City, Utah*, 2010.

[3] A. J. Kim, A. A. Lazar, and Y. Slutskiy, "System identification of dm4 glomerulus in the drosophila antennal lobe using stationary and non-stationary odor stimuli," *BMC Neuroscience*, vol. 11, no. Suppl 1, p. P174, 2010.

[4] A. J. Kim, A. A. Lazar, and Y. B. Slutskiy, "System identification of drosophila olfactory sensory neurons," *Journal of computational neuroscience*, vol. 30, no. 1, pp. 143–161, 2011.

[5] R. I. Wilson, "Understanding the functional consequences of synaptic specialization: insight from the drosophila antennal lobe," *Current opinion in neurobiology*, vol. 21, no. 2, pp. 254–260, 2011.

[6] S. A. Budick and M. H. Dickinson, "Free-flight responses of drosophila melanogaster to attractive odors," *Journal of experimental biology*, vol. 209, no. 15, pp. 3001–3017, 2006.

[7] A. Y. Katsov and T. R. Clandinin, "Motion processing streams in drosophila are behaviorally specialized," *Neuron*, vol. 59, no. 2, pp. 322–335, 2008.

[8] G. Maimon, A. D. Straw, and M. H. Dickinson, "A simple vision-based algorithm for decision making in flying drosophila," *Current Biology*, vol. 18, no. 6, pp. 464–470, 2008.

[9] M. E. Chiappe, J. D. Seelig, M. B. Reiser, and V. Jayaraman, "Walking modulates speed sensitivity in drosophila motion vision," *Current Biology*, vol. 20, no. 16, pp. 1470–1475, 2010.

[10] D. B. Chklovskii, S. Vitaladevuni, and L. K. Scheffer, "Semi-automated reconstruction of neural circuits using electron microscopy," *Current opinion in neurobiology*, vol. 20, no. 5, pp. 667–675, 2010.

[11] S.-y. Takemura, A. Bharioke, Z. Lu, A. Nern, S. Vitaladevuni, P. K. Rivlin, W. T. Katz, D. J. Olbris, S. M. Plaza, P. Winston *et al.*, "A visual motion detection circuit suggested by drosophila connectomics," *Nature*, vol. 500, no. 7461, pp. 175–181, 2013.

[12] S. R. Olsen and R. I. Wilson, "Cracking neural circuits in a tiny brain: new approaches for understanding the neural circuitry of Drosophila," *Trends in neurosciences*, vol. 31, no. 10, pp. 512–520, 2008.

[13] A.-S. Chiang, C.-Y. Lin, C.-C. Chuang, H.-M. Chang, C.-H. Hsieh, C.-W. Yeh, C.-T. Shih, J.-J. Wu, G.-T. Wang, Y.-C. Chen *et al.*, "Three-Dimensional Reconstruction of Brain-wide Wiring Networks in Drosophila at Single-Cell Resolution," *Current Biology*, vol. 21, no. 1, pp. 1–11, 2011.

[14] P. Gleeson, S. Crook, R. C. Cannon, M. L. Hines, G. O. Billings, M. Farinella, T. M. Morse, A. P. Davison, S. Ray, U. S. Bhalla *et al.*, "NeuroML: a language for describing data driven models of neurons and networks with a high degree of biological detail," *PLoS computational biology*, vol. 6, no. 6, p. e1000815, 2010.

[15] N. T. Carnevale and M. L. Hines, *The NEURON book*. Cambridge University Press, 2006.

[16] D. Beeman, "GENESIS Modeling Tutorial," *Brains, Minds, and Media*, vol. 1, 2005.

[17] R. M. Cubert and P. A. Fishwick, "MOOSE: an object-oriented multimodeling and simulation application framework," *Simulation*, vol. 6, 1997.

[18] M.-O. Gewaltig and M. Diesmann, "Nest (neural simulation tool)," *Scholarpedia*, vol. 2, no. 4, p. 1430, 2007.

[19] D. S. Chevitarese, L. E. Givon, A. A. Lazar, and M. Vellasco, "CircuitML: a Modular Language for Modeling Local Processing Units in the Drosophila Brain," in *Frontiers Neuroinformatics*. Frontiers Research Foundation, Jul. 2013, pp. 80–81.

[20] L. E. Givon and A. A. Lazar, "An open architecture for the massively parallel emulation of the Drosophila brain on multiple GPUs," *BMC Neuroscience*, vol. 13, pp. 1–2, 2012.

[21] G. Booch, "Object-oriented development," *Software Engineering, IEEE Transactions on*, no. 2, pp. 211–221, 1986.

[22] A. Kamikouchi, "Auditory neuroscience in fruit flies," *Neuroscience research*, vol. 76, no. 3, pp. 113–118, 2013.

[23] H. T. Spieth, "Courtship behavior in drosophila," *Annual review of entomology*, vol. 19, no. 1, pp. 385–405, 1974.

[24] A. Bretman, J. D. Westmancoat, and T. Chapman, "Male control of mating duration following exposure to rivals in fruitflies," *Journal of insect physiology*, vol. 59, no. 8, pp. 824–827, 2013.

[25] M. J. Kernan, "Mechanotransduction and auditory transduction in drosophila," *Pflügers Archiv-European Journal of Physiology*, vol. 454, no. 5, pp. 703–720, 2007.

[26] M. C. Göpfert and D. Robert, "The mechanical basis of drosophila audition," *Journal of Experimental Biology*, vol. 205, no. 9, pp. 1199–1208, 2002.

[27] F. von Schilcher, "The role of auditory stimuli in the courtship of drosophila melanogaster," *Animal Behaviour*, vol. 24, no. 1, pp. 18–26, 1976.

[28] K.-F. Fischbach and A. Dittrich, "The optic lobe of drosophila melanogaster. i. a golgi analysis of wild-type structure," *Cell and tissue research*, vol. 258, no. 3, pp. 441–475, 1989.

[29] Y. Zhu, A. Nern, S. L. Zipursky, and M. A. Frye, "Peripheral visual circuits functionally segregate motion and phototaxis behaviors in the fly," *Current Biology*, vol. 19, no. 7, pp. 613–619, 2009.

[30] S. J. Caron, V. Ruta, L. Abbott, and R. Axel, "Random convergence of olfactory inputs in the Drosophila mushroom body," *Nature*, 2013.

[31] L. B. Vosshall and R. F. Stocker, "Molecular architecture of smell and taste in Drosophila," *Annu. Rev. Neurosci.*, vol. 30, pp. 505–533, 2007.

[32] C. Koch and I. Segev, *Methods in neuronal modeling: from ions to networks*. MIT press, 1998.

[33] R. S. Zucker and W. G. Regehr, "Short-term synaptic plasticity," *Annual review of physiology*, vol. 64, no. 1, pp. 355–405, 2002.

[34] E. A. Hallem and J. R. Carlson, "Coding of odors by a receptor repertoire," *Cell*, vol. 125, no. 1, pp. 143–160, 2006.