Реконструкція мереж генної регуляції у Грід-середовищі за допомогою nordugrid-arc-python API

Аліна Фролова, Марія Оболенська

Інститут молекулярної біології і генетики НАН України, вул. Академіка Заболотного, 150, Київ, Україна

fshodan@gmail.com, m.obolenska@gmail.com

Резюме. *Розвиток нових технологій обумовив появу міждисциплінарної галузі в біології – системної біології, метою якої є найбільш повне системне відображення біологічних процесів на підставі аналізу величезної кількості даних. Один із підходів в цій галузі – це моделювання мережі генної регуляції, яке надає найповніше уявлення про функціонування клітин/органів і стрімко набуває поширення в медицині та біології. Моделювання передбачає поєднання високотехнологічних підходів в галузі молекулярної біології і інформатики, а саме широкомасштабного дослідження експресії генів за умов повторюваних збурень системи і моделювання мережі за використання найсучасніших методів інформаційних технологій, таких як розподілені обчислення. Оскільки сучасні алгоритми реконструкції не в змозі побудувати генні мережі з тисяч генів за прийнятний час, ми пропонуємо використання грід-технологій для вирішення цієї задачі з розробкою автоматичної системи менеджменту задач на основі nordugrid-arc-python API.*

Ключові слова

Розподілені обчислення, мережі генної регуляції, булеві мережі, баєсові мережі, теорія інформації, грід, nordugrid-arc-python.

1 Вступ

Впродовж другої половини XX сторіччя розвиток молекулярної біології означився дослідженням високомолекулярних сполук, задіяних у збереженні, передачі і реалізації генетичної інформації, але кількість цих сполук була обмежена можливостями тогочасних технологій.

Цей редукціоністський підхід приніс і ще приносить багато відкриттів і нових результатів, які використовуються і можуть бути використані у практиці – генетично модифіковані рослини, рекомбінантні білки, які замінили виділені природні білки в медичній практиці (наприклад, рекомбінантний інсулін, інтерферон, ростові фактори і багато інших), тест-системи і нові ліки. Водночас стало зрозумілим, що біологічні системи працюють як розгалужена мережа сполук, що взаємодіють між собою і зумовлюють появу «нової якості» в системі (emergency property), яку неможливо осягнути, досліджуючи окремі компоненти системи [1]

Розвиток нових технології обумовив появу нової галузі в біології - системної біології, метою якої є найбільш повне системне відображення біологічних процесів на підставі аналізу величезної кількості даних. Один із підходів в цій галузі – це моделювання мережі генної регуляції, яке надає найповніше уявлення про функціонування клітин/органів і стрімко набуває поширення в медицині та біології.

Створюються генні мережі різних процесів (метаболічних, циркадних ритмів, програмованої загибелі клітин, регуляції противірусної відповіді, онкозахворювань, наприклад передміхурової залози). І цей підхід вже дає свої перші результати – підвищується продуктивність мікроорганізмів з напрацювання певних речовин, з'являються нові мішені для лікування хвороб, зокрема передміхурової залози [2]. На пострадянському просторі в Сибірському відділенні РАН виконується масштабний міждисциплінарний проект «Моделювання фундаментальних генетичних процесів і систем», в якому беруть участь шість провідних закладів (ІЦіГ, ІМ, Інститут обчислювальних технологій, Інститут обчислювальної математики і математичної геофізики, ІТФ і Міжнародний Томографічний Центр).

На сьогодні створені у відділі системної біології СВ РАН генні мережі включають близько 50 генів, 200 білків, між якими встановлено більше 500 взаємодій (http://wwwmgs.bionet.nsc.ru/mgs/gnw/genenet/ GeneNetfunctional sections). За сучасними оцінками в геномі людини налічується до 25 – 30 тис генів, які взаємодіють між собою в різних варіантах і функціонують в різних режимах. Це дає уяву про надзвичайно великий розмір генетичної мережі в організмі людини і зумовлює складність у розв'язанні цієї проблеми .

Простір пошуку оптимальної мережі генної регуляції є супер-експоненційним: для 9 генів маємо $1.21*10^{15}$ можливих варіантів, 20 генів – $2.34*10^{72}$, 30 генів – $2.71*10^{158}$ [3]. Тому всі спроби моделювання обмежені використанням евристик, що зменшують поле пошуку (алгоритми імітації відпалу, жадібні алгоритми, генетичні алгоритми), але водночас призводять до падіння точності моделі мережі регуляції (Min Zou et al, 2005). Особливо актуальною проблема падіння точності стає при використанні реальних біологічних даних (замість тренувальних моделей), оскільки їхня кількість обмежена, що спричинено великою вартістю одержання цих даних. Однак навіть ті алгоритми, що використовують евристики не здатні реконструювати великі мережі (порядку тисяч генів) за прийнятний час.

Тому наші зусилля були спрямовані на розробку зручної системи обчислень на GRID, яка б допомогла автоматизувати процес реконструкції мереж генної регуляції та зробити можливим побудову мереж з багатьох тисяч генів.

2 Теоретична частина

Мережа генної регуляції – це сукупність опосередковано пов'язаних між собою модульних елементів ДНК (генів), які приймають множинні вхідні сигнали у вигляді РНК і білків, обробляють сигнали і зумовлюють темп, з яким гени мережі транскрибуються в РНК і транслюються в білки. Архітектура мережі віддзеркалює взаємодію її різних елементів і дає найповнішу уяву щодо регуляції функціонування клітини на відміну від традиційного вивчення поодиноких генів, тому реконструкція генних мереж є важливим предметом вивчення системної біології.

На разі використовуються близько десяти підходів до моделювання генних мереж: машинне навчання, баєсові мережі, булеві мережі, диференційні рівняння, теорія інформації, мережі Петрі, нейронні мережі, генетичні алгоритми [4, 5, 6]. Кожен з цих підходів має свої переваги та недоліки, визначення яких ускладнене браком ґрунтовних оглядів в науковій літературі. Іншою проблемою є те, що згадані підходи застосовуються для реконструкції невеликих мереж (порядку сотень генів).

Зі збільшенням кількості генів обчислювальна складність зростає експоненційно: для 30 генів вже маємо $2.71*10^{158}$ можливих варіантів мереж у випадку використання баєсових мереж [3], хоча для теорії інформації маємо значно меншу оцінку складності [7]. Однак, задача побудови генних мереж все одно є NP-повною [4], тому її неможливо вирішити без застосування розпаралелювання на розподілених обчислювальних системах, таких як грід.

Іншою проблемою побудови генних мереж є їх біологічна релевантність, тобто те, наскільки точно вони відображають біологічні процеси, "зріз" яких отримано під час експерименту. Використання евристик безумовно зменшує поле пошуку, але водночас призводять до падіння точності моделі мережі регуляції, що є критичним при використанні реальних біологічних даних (замість тренувальних моделей), оскільки їхня кількість обмежена, що спричинено великою вартістю одержання цих даних. Тому використання розподілених обчислень є критичним при вирішенні даної проблеми.

Існує декілька моделей, які покладені в основу побудови генних мереж. Найбільш вживані – це булеві мережі, баєсові мереж, клас ймовірнісних графових моделей та звичайні диференційні рівняння. Для кожної з цих моделей можна застосувати різні методи, які у свою чергу можна реалізувати різними алгоритмами. Однак лише деякі з комбінацій модель-метод-алгоритм можуть бути застосовані для реконструкції великих мереж у розподіленому середовищі й при цьому надавати адекватну інформацію щодо архітектури вихідної мережі. Отже, нами було виокремлено такі недоліки [8]:

Булеві мережі: дискретні значення станів генів; детерміністична динаміка мереж, що залежить від початкового стану.

Баєсові мережі (динамічні): обчислювально складні; проблема локальних максимумів.

Диференційні рівняння: залежать від параметрів, які складно отримати експериментально; важко вирішувати.

Теорія інформації: визначає тільки рівень взаємодії пар генів (для напрямку взаємодії потрібно застосовувати окремі методи).

Свій вибір ми зупинили переважно на методах, які засновані на **теорії інформації**. В основу їх покладено поняття диференціальної ентропії та спільної інформації, які можна визначити наступним чином:

Диференціальною ентропією $h(X)$ випадкової неперервної величини X з щільністю $f(x)$ називається: $h(X) = -\int_S f(x)\log f(x)dx$, де S - підтримуюча множина випадкової величини [9].

Тоді спільна інформації двох неперервних випадкових величин Xта Y:
$M(X,Y) = \int\int f(x,y)\log\left(\frac{f(x,y)}{f(x)\cdot f(y)}\right)dxdy$, де $f(x,y)$- сумісна ймовірнісна щільність двох випадкових величин, а $f(x)$та $f(y)$- граничні ймовірності.

Але ці ймовірнісні функції є невідомими і повинні визначатися з вимірювань, що є дуже складною задачею. Одним з методів обчислення спільної інформації є обчислення щільності ядра (kernel density estimation), яке було запропоноване Муном та ін. та має низку переваг на гістрограмним методом [10]:

1. Краща середньоквадратична похибка збіжності обчислень відносно щільності.
2. Нечутливість до вибору початкової точки.
3. Можливість визначати більш складні форми, ніж прямокутні стовпці в гістограмах.

Програмні пакети Aracne [7], CLR [11], MRNet [12], що ми обрали, використовують саме такий підхід.

Іншим підходом, що ми обрали є **динамічні баєсові мережі**, які можна розглядати як розширення звичайних баєсових мереж і які здатні відобразити динаміку генних мереж. Нехай змінна серійних даних мікромасив-експерименту $x \in R^{n\times p}, x_{ti}$, де n- кількість часових точок, а p- кількість генів, позначає спостереження гена iу час t, тоді вектор спостереження у час tможна представити у вигляді $x_{(t)} = \left[x_{t1},\dots,x_{tp}\right]^T$, а $i-$йген у всіх часових точках - $x_{(i)} = [x_{1i},\dots,x]^T$. Динамічні баєсові мережі припускають залежність від часу, в якій направлені ребра повинні "рухатися" вперед з часом [13].

Сумісний ймовірнісний розподіл для динамічної баєсової мережі може бути обчислений як:
$$P\left(x_{11},\dots,x_{np}\right) = \prod_{i=1}^{p}\prod_{t=1}^{n}\left(P\left(x\right)_{ti} \mid L(x_{ti})\right)$$
$p\left(x_i \mid L(x_i)\right)$- умовний ймовірнісний розподіл визначений для кожного x_i, де $L(x_i)$– змінна, що відноситься до прямих регуляторів i

Зазвичай припускають, що такі мережі — це моделі Маркова першого порядку, в яких на кожен ген безпосередньо впливають лише попередні гени [14]. Оскільки ці моделі залежні від часу, то легко побудувати мережу з циклами зворотнього зв'язку

Програмний інструмент Banjo [15] виявився єдиним, який здатен реконструювати великі мережі, реалізуючи при цьому підхід на основі динамічних баєсових мереж (знаходження топології мережі відбувається методом відпалу), однак, він потребує багато оперативної пам'яті, що є недоліком. Хоча при цьому Banjo має вбудовану можливість розпаралелення на декілька потоків. Динамічні баєсові мережі здатні визначити напрямок взаємодії генів, на відміну від методів основаних на теорії інформації, тому підвищують точність реконструкції генних мереж у поєднання з теорією інформації.

Для того, щоб перевірити точність реконструкції обраних програмних пакетів ми використовували декілька різних видів даних: **"золоті стандарти" та синтетичні мережі.**

"Золоті стандарти" були взяті з даних проекту DREAM Challenge (Dialogue for Reverse Engineering Assessments and Methods http://www.the-dream-project.org/), який якраз має на меті порівняння різних чисельних методів у галузі системної біології. Проект надає дані експресії генів і відповідні ним готові структури генних мереж, а також програмні засоби (скрипти Matlab) для оцінки подібності власне отриманих результатів з результатми «золотого стандарту».

Фактично було використано дані п'ятої задачі конкурсу DREAM-2 (2007 рік), які складалися з 300 мікромасив-експериментів, у кожному з яких було визначено експресію 3456 генів. Мікромасив-експерименти були проведені з використанням РНК кишкової палички (*Escherichia coli*), геном якої (4 639221 пар основ) містить 4377 генів.

Іншим типом даних, на яких ми перевіряли точність реконструкції, були синтетичні мережі, тобто програмно згенеровані дані експресії генів для яких також заздалегідь відома структура вихідної мережі. Для генерації синтетичних мереж ми використали GeneNetWeaver (http://gnw.sourceforge.net). Топологія мереж була отримана шляхом вилучення підмереж з транскрипційної регуляторної мережі E.Coli. Динаміка мереж симульована з детальної кінетичної моделі генної регуляції за допомогою стохастичних диференційних рівнянь з додаванням шуму. Параметри отриманих мереж (кількість збурень, часові точки та ін.) відповідають DREAM-4 InSilico Challenge (http://gnw.sourceforge.net/dreamchallenge.html#dream4challenge). Було згенеровано мережі різної розмірності: 10 генів (10 регуляторів, 14 взаємодій), 100 генів (100 регуляторів, 398 взаємодій), 1000 (100 регуляторів, 4350 взаємодій), 157 генів (усі регулятори з регуляторної мережі E.Coli, 490 взаємодій).

Використані в роботі дані можна знайти у відкритому репозиторії https://github.com/sysbio-vo/grn-grid-data

Оцінка методів реконструкції відбувалася за декількома критеріями: здатність відтворювати конкретні структурні мотиви, площа під кривою точності-пам'яті (area under receiver operating characteristic), площа під кривою чутливості-специфічності (area under precision-recall).

У цілому методи, засновані на теорії інформації показали кращі результати, ніж динамічні баєсові системи. Однак, ці методи найбільш чутливі до даних з декількома видами збурень (multiperturbed data), тож вони показали не досить хороший результат на синтетичних даних експресії генів в часовій послідовності (time-series experiment).

Для реконструкції баєсових динамічних мереж використовувався алгоритм відпалу та було встановлене часове обмеження роботи алгоритму, оскільки на відміну від теорії інформації, де послідовно обчислюється спільна інформація усіх пар генів, у данному випадку алгоритм на кожному кроці намагається вдосконалити архітектуру баєсової мережі, тому немає як такого завершення, окрім як встановлення порогу для функції оцінки або часу виконання алгоритму.

Однак, Banjo дозволяє налаштовувати багато вхідних параметрів алгоритма, тому ще залишається простір для експериментів з даним методом.

Таб. 1. Результати тестування 4 інструментів реконструкції мереж генної регуляції.

Tool	AUPR (area under precision-recall curve)	AUROC (area under receiver operating characteristic curve)
ARACNE	0.065	**0.665**
MRNET	0.015	0.611
CLR	0.072	0.577
Banjo	**0.111**	0.563

Для реконструкції мереж генної регуляції ми використовували **українську грід-інфраструктуру**, а саме кластери arc.imbg.org.ua, cluster.immsp.kiev.ua, ds4.ilt.kharkov.ua, arc.univ.kiev.ua, golowood.mao.kiev.ua, grid.isma.kharkov.ua, arc-emi.bitp.kiev.ua, vobox-edu.bitp.kiev.ua.

Запуск задач здійснювався скриптами, написаними мовою python, на основі **nordugrid-arc-python 2.0.1**, що дозволило автоматизувати процес реконструкції генних мереж, оскільки цей python wrapper бібліотек arc дозволяє більш гнучко використовувати функціонал API.

Python інтерфейс до arclib версії 2.0.1 відрізняється від більш старих версій (приклади використання попереднього інтерфейсу можна знайти за даним посилання http://wiki.nordugrid.org/index.php/ARC_Compute_Clients/Python_examples), після значної реструктуризації C++ бібліотеки надають більш дружній інтерфейс, також модель даних більш подібна на модель GLUE2 (http://www.nordugrid.org/arc/releases/12.05/release_notes_12.05.html). Щодо знаходження обчислювальних ресурсів класи TargetGenerator та TargetRetriever були замінені на ServiceEndpointRetriever, TargetInformationRetriever, ComputingServiceRetriever та JobListRetriever класи, де ServiceEndpointRetriever повинен використовуватися для запиту індексу сервісів реєстрації для будь-якого типу сервісу, TargetInformationRetriever для отримання інформації щодо локальних сервісів, ComputingServiceRetriever об'єднує функіональність перших двох, та JobListRetriever використовується для отримання інформації про задачі. Причина цих змін полягає в тому, що TargetGenerator та TargetRetriever класи використовувались для різного виду функціоналу одночасно та не були досить гнучкими.

Отже, створення endpoint для обчислювального елементу тепер виглядає так:

```
    uc = arc.UserConfig()

    # Creating an endpoint for a Computing Element
    endpoint = arc.Endpoint("cluster.immsp.kiev.ua",
arc.Endpoint.COMPUTINGINFO)

    retriever = arc.ComputingServiceRetriever(uc, [endpoint])
    retriever.wait()
```

```
target = retriever.GetExecutionTargets()
```

Ми можемо не зазначати InterfaceName для отримання інформації про обчислювальний елемент (org.nordugrid.ldapng, org.nordugrid.ldapglue2, org.nordugrid.wsrfglue2, org.ogf.emies), API автоматично спробує кожен з інтерфейсів. Також для опису завдання не є необхідним використання (але є можливим) xrsl та jsdl файлів, оскільки nordugrid-arc-python API дозволяє створювати завдання динамічно, в залежності від поточної задачі, наприклад:

```
# Create a JobDescription
jobdesc = arc.JobDescription()
jobdesc.Application.Executable.Path = "aracne2"
jobdesc.Application.Output = "stdout.txt"
```

Отже запуск задачі відбувається таким чином:

```
uc = arc.UserConfig()
job = arc.Job()
target.Submit(uc, jobdesc, job)
```

Для того, щоб забрати результати виконання задач, ми можемо зберегти відомості про кожну з задач:

```
job.WriteJobsToFile("/home/aln/.arc/jobs.xml", [job])
```

або ж отримати перелік усіх задач з конкретного обчислювального елементу:

```
# Creating a container which will store the retrieved jobs
jobs = arc.JobContainer()
# Create a job list retriever
retriever = arc.JobListRetriever(uc)
# Add our container as the consumer of this retriever, so it will get the
results
retriever.addConsumer(jobs)
# Add our endpoint to the retriever, which starts querying it
retriever.addEndpoint(endpoint)
# Wait until it finishes
retriever.wait()
```

Завантаження виконаних задач здійснюється таким чином:

```
# Put the job into a JobSupervisor and update its information
job_supervisor = arc.JobSupervisor(uc, [job])
job_supervisor.Update()
# Prepare a list for storing the directories for the downloaded job results
(if there would be more jobs)
downloadeddirectories = arc.StringList()
# Start retrieving results of all the selected jobs
success = job_supervisor.Retrieve("/tmp", False, False,
downloadeddirectories)
```

Оскільки усі наявні керівництва по застосуванню nordugrid-arc-python API висвітлюють лише старі версії, робочі скрипти для запуску і моніторингу задач, які можна знайти у відкритому репозиторії https://github.com/sysbio-vo/grn-grid-project, можуть слугувати базисом для дослідників, які будуть користуватися новою версією API.

Грід middleware наразі активно розвивається, тому використання нового програмного забезпечення з одного боку значно пришвидшує розробку і застосування прикладних проектів у грід, з іншого — надає розробникам цього забезпечення відгук для виправлення помилок та вдосконалення функціоналу. Нами було реконструйовано генну мережу з даних експресії 10455 генів (8000 експериментів) гепатоцитів щура (http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE8858). Задачу було поділено на 2091 частину, виконання кожної з яких займає близько 400 хвилин на Intel(R) Xeon(R) CPU E5520 @ 2270 MHz, що приблизно становить 580 днів послідовних обчислень. Обчислення ж на гріді на 6 різних кластерах зайняли 15 днів з урахуванням часу на запуск та забір задач, що безсумнівно доводить доцільність та зручність використання грід-технологій для вирішення проблем системно біології.

3 Висновки

Метою нашої роботи було створення автоматичної системи реконструкції мереж генної регуляції, які налічують багато тисяч генів, за прийнятний час та з високою біологічною достовірністю. Оскільки складність пошуку оптимальної мережі експоненційна, застосування грід-технологій є невід'ємною частиною реалізації цієї мети.

Нами було здійснено аналіз багатьох методів реконструкції та обрано найкращі з них. Це динамічні баєсові мережі та методи, засновані на теорії інформації. Визначення точності реконструкції кожного з методів за допомогою обраних на основі проведеного аналізу алгоритмів (Aracne, Banjo, CLR, MrNet).

Для побудови системи менеджменту задач в грід-середовищі ми використали nordugrid-arc-python 2.0.1 API, що якнайкраще підходить для наших цілей, оскільки дозволяє автоматизувати усі ланки процесу роботи з задачами без написання скриптів-обгорток для arc client. На основі побудованої системи було реконструйовано генну мережу з близько 10000 генів та показано доцільність виокристання грід-технологій для вирішення проблем побудови регуляторних мереж.

4 Подяки

Дана робота виконувалась у рамках Державної цільової науково-технічної програми впровадження і застосування грід-технологій на базі ІБМГ НАНУ (проекти 2010 і 2011р.р. № Г16- 46 і 2012р.№ 69-53).

Посилання

[1] Hartwell L.H., Hopfield J.J., Leibler S., Murray A.W. From molecular to modular cell biology. Nature. 1999 Dec 2;402(6761 Suppl):C47-52.

[2] Bonnet E., Michoel T. and Van de Peer Y. Prediction of a gene regulatory network linked to prostate cancer from gene expression, microRNA and clinical data. Bioiformatics Vol. 26 ECCB 2010, pages i638–i644

[3] Ott S., Imoto S., Miyano S. Finding optimal models for small gene networks. *Pacific Symposium on Biocomputing*. P. 557-567, 2004.

[4] Lee Wei-Po, Wen-Shyong Tzou. Computational methods for discovering gene networks from expression data // Brief Bioinform.–2009.– Vol.10,N 4.–P.408-423.

[5] Hecker M., Lambeck S., Toepfer S., et al. Gene regulatory network inference: data integration in dynamic models— a review // Biosystems.–2009.–Vol. 96.–P. 86-103.

[6] Karlebach G, Shamir R. Modelling and analysis of gene regulatory networks // Nat Rev Mol Cell Biol.–2008.– Vol. 9.–P. 770-80.

[7] Margolin A.A, Nemenman I., Basso K., Wiggins C., Stolovitzky G, Favera R.D, Califano A. ARACNE: An Algorithm for the Reconstruction of Gene Regulatory Networks in a Mammalian Cellular Context. *BMC Bioinformatics*. Vol. 7(S7), 2006.

[8] Frolova A.O. Overview of methods of reverse engineering of gene regulatory networks: Boolean and Bayesian networks. Biopolymers and Cell. Vol. 28. N 3, 2012.

[9] Cover T. M. and J. A. Thomas, Elements of information theory. John Wiley and Sons, 2006.

[10] Moon Y.-I., B. Rajagopalan, and U. Lall, "Estimation of mutual information using kernel density estimators," Physical Review E, vol. 52, no. 3, p. 2318, 1995

[11] Faith J.J., B. Hayete, J.T. Thaden, I. Mogno, J. Wierzbowski, G Cottarel, Kasif S., J.J. Collins, and T.S. Gardner. Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. PLoS Biology, 5, 2007.

[12] Meyer, P. E., Kontos, K., Lafitte, F., & Bontempi, G (2007). Information-theoretic inference of large transcriptional regulatory networks. EURASIP Journal on Bioinformatics and Systems Biology, 2007.

[13] Wu H., Liu X. Dynamic bayesian networks modeling for inferring genetic regulatory networks by search strategy: Comparison between greedy hill climbing and mcmc methods // Proceedings of World Academy of Science, Engineering and Technology.–2008.–Vol. 34.–P. 224–234.

[14] Sima C., Hua J., S. Jung S. Inference of Gene Regulatory Networks Using Time-Series Data: A Survey // Current Genomics. –2009.–Vol. 10, N. 6.–P. 416-429.

[15] Smith, V., Yu, J., Smulders, T., Hartemink, A., & Jarvis, E. (2006) "Computational Inference of Neural Information Flow Networks." PLoS Computational Biology, 2, November 2006.

[16] Dietterich, T. G: Ensemble Methods in Machine Learning. Lecture Notes in Computer Science, 1857 (2000) 1-15

[17] Breiman, Leo. "Stacked regressions." Machine learning 24.1 (1996): 49-64.

[18] Snyman Jan A. (2005). Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms. Springer Publishing. ISBN 0-387-24348-8