Журнал обчислювальної та прикладної математики | 2015, №1(118) | Journal of Computational & Applied Mathematics

УДК 517.9

ФРАГМЕНТАЛЬНИЙ ПІДХІД ДО РОЗВ'ЯЗАННЯ ЗАДАЧІ ПРОГНОЗУВАННЯ ТРЕТИННОЇ СТРУКТУРИ МОЛЕКУЛИ ПРОТЕЇНУ

В. О. Рудик

Резюме. В роботі запропоновано популяційний алгоритм для розв'язання задачі прогнозування третинної структури протеїнів. Специфікою алгоритму є фрагментарний підхід до побудови структур молекул, що дозволяє проводити пошук в просторі допустимих розв'язків уникаючи недопустимих варіантів з врахуванням особливостей задачі. Ефективність розглянутої процедури підвищується за рахунок використання у ній схем локального пошуку.

Вступ

Проблема прогнозування третинної структури молекули протеїну актуальна в сферах молекулярної біології, біохімії, генетики, фармакології. Можливість передбачувати структуру протеїнів є ключем до розуміння їх впливу на організм, схем виникнення багатьох хвороб, і часто знаходження шляхів їх лікування. В математичному плані зазначена проблема у більшості випадків моделюється задачею пошуку такої ламаної лінії (згортки) без самоперетину, яка проходить по вузлам вибраної решітки і доставляє мінімум цільовій функції задачі.

Основним правилом, яким керуються при дослідженні білків, є доктрина "Послідовність-структура-функціональність". Вважається, що функції, які виконуються тим чи іншим білком, залежать від його форми у просторі — третинної структури. Третинна структура ж, в свою чергу, однозначно визначається будовою білка — послідовністю амінокислот, що являє собою первинну структуру протеїну. Однією з моделей, що використовується для прогнозування форми молекули протеїну математичними методами є гідрофобно-полярна модель Ділла [1]. Відповідно до неї амінокислоти в залежності від своїх властивостей поділяються на два класи — гідрофобні і полярні. Первинна структура подається у формі послідовності елементів з множини {H, P}, де H позначає гідрофобну амінокислоту, а P — полярну. Структура молекули подається у вигляді шляху без самоперетинів у деякій дискретній решітці, кожна амінокислота розташована в певному вузлі решітки, сусідні в послідовності амінокислоти — в сусідніх вузлах. Вважається, що однією з основних сил, які впливають на формування структури молекули, є зв'язки (H-H контакти), що утворюються між гідрофобними залишками, розташованими поруч, в дискретній моделі — в сусідніх вузлах

решітки. Енергія структури в моделі Ділла підраховується як кількість H-H контактів у ній зі знаком мінус. Вважається, що молекула прийме ту форму, в якій її енергія мінімальна. Таким чином постає задача комбінаторної оптимізації. Доведено, що ця задача є NP-складною [2].

Для розв'язання задачі прогнозування третинної структури молекули протеїну застосовувались алгоритми мурашиних колоній [3, 4], генетичні алгоритми [5, 6], алгоритми детермінованого та стохастичного локального пошуку [7, 8]. Специфіка задачі полягає в тому, що умова допустимості згортки, а саме відсутність в ній самоперетинів, є достатньо жорсткою. При цьому, враховуючи, що структури з низькою енергією є достатньо компактними за рахунок великої кількості гідрофобних зв'язків у них, кількість допустимих згорток в їх околі є малою. Проте, можна показати, що множина допустимих розв'язків з введеним поняттям околу у ній є зв'язною.

За показником співвідношення оптимальності знайденого розв'язку до часу, затраченого на його пошук, ефективними виявились алгоритми локального пошуку. Зокрема це пояснюється тим, що вони, на відміну від алгоритмів, в яких використовується послідовна процедура побудови згортки, не приходять до тупикових варіантів. Це спонукає до розробки алгоритму, який буде максимально використовувати в собі процедуру локального пошуку, але при цьому застосовувати диверсифікацію пошуку для аналізу усього простору допустимих розв'язків.

2. Опис алгоритму

Алгоритм, що пропонується в роботі, є популяційним, тобто він на кожній ітерації оперує відразу з декількома розв'язками. Ідея полягає в тому, що на кожному кроці проводиться пошук у околі більш перспективних (відповідно до значення фітнес-функції) розв'язків. Для цього буде використовуватись відбір з повтореннями. Таким чином згортки з більш низьким значенням енергії з високою ймовірністю потраплять в нову популяцію у декількох копіях, кожна з яких на заданій ітерації замінюється на іншу згортку з її околу з достатньо низьким значенням енергії (таку заміну надалі будемо називати мутацією, процедура мутації буде розглянута нижче). Якщо ж такої структури не знайдено, проводиться диверсифікація пошуку — задана структура замінюється на іншу, яка генерується випадково.

У такій схемі важливим є адекватне визначення фітнес-функції. Адже якщо її значення буде залежати лише від енергії згортки, нові структури будуть витіснені тими, що були побудовані в результаті більшої кількості ітерацій. Це спонукає до введення поняття віку згортки, яке визначимо наступним чином: при додаванні випадкової структури в популяцію її вік вважатимемо 0; при мутації вік мутованої структури збільшується на одиницю. При відборі елементів у нову популяцію у фітнес-функції враховутиметься не тільки енергія структури, але й її вік.

Ще однією особливістю алгоритму є процедура мутації структури. Тут прийнято до уваги той факт, що чим довша амінокислотна послідовність, тим більша в її околі частка недопустимих структур. Тому при побудові структури доречно формувати її фрагментами, позиціонуючи в решітці

спочатку центральні елементи, потім поступово додавати решту. На першій ітерації популяція заповнюється випадковими згортками фіксованої довжини, меншої, ніж довжина амінокислотної послідовності. Далі, коли задана частина згортки буде побудована, до всіх елементів популяції з обох сторін додається по одному амінокислотному залишку в випадковому напрямку. Для ефективного пошуку структур з низьким значенням енергії в алгоритмі використовується Гауссівський розподіл, параметри якого змінюються від ітерації до ітерації. Цей розподіл задає ймовірність, з якою заданий елемент згортки буде змінено в процесі мутації. Параметри розподілу змінюються таким чином, що ймовірність вибору центральних елементів у згортці зменшується при збільшенні ітерації. Сама ж процедура мутації полягає в пошуку структури з нижчою енергією в околі заданого варіанту згортки.

3. Схема алгоритму

Обчислювальна схема алгоритму подана на рис. 1. На першому кроці алгоритму процедурою *ІніціалізуватиПараметриРозподілу*() встановлюються параметри розподілу, що буде використовуватись алгоритмом при мутації розв'язків. Розподіл задаватиме ймовірність, з якою відповідний елемент згортки буде змінюватись. У якості функції щільності розподілу пропонується взяти нормовану суму двох функцій щільності Гауссівського розподілу з середніми значеннями, симетричними відносно середини молекули:

$$\bar{f}(x) = 0.5(f(x, n/2 - M, \sigma) + f(x, n/2 + M, \sigma)$$

де M та σ — параметри розподілу, а f позначає функцію щільності Гауссівського розподілу. Початкове значення для M встановлюється рівним 0, початкове значення σ є параметром алгоритму. Параметри M та σ будуть змінюватись в процесі роботи процедури.

Ще одним параметром алгоритму є довжина фрагментів згорток l, що будуються на першій ітерації. Відповідно до правила трьох сигм, індекси, що генеруватимуться відповідно до введеного розподілу, з ймовірністю, більшою 0.997 будуть лежати в діапазоні $(n/2 - 3\sigma, n/2 + 3\sigma)$, тому доречно обрати $l = 6\sigma$.

Наступним кроком алгоритму є генерація початкової популяції. Процедура *ЗгенеруватиПочатковуПопуляцію*() випадковим чином генерує популяцію згорток довжиною l.

Оптимальне знайдене значення розв'язку будемо позначати символом s.

Для кожного розв'язку в популяції зберігатиметься його вік — кількість ітерацій, впродовж яких цей розв'язок існує в популяції. На першому кроці вік кожного розв'язку встановлюється рівним 0.

Далі запускається ітераційна процедура, яка зупиниться, коли буде виконуватись умова завершення. Процедура *ВиконуєтьсяУмоваЗавершення*() може базуватись на кількості кроків, впродовж яких оптимальний розв'язок не змінювався, але при цьому має виконуватись рівність $l = n$. При умові $l < n$ згортка, побудована алгоритмом, не буде повною.

На кожній ітерації алгоритму обчислюється вага всіх елементів популяції: чим нижче значення енергії розв'язку, тим вища його вага. В розробленому алгоритмі пропонується обрати вагу рівною значенню енергії зі знаком мінус.

```
procedure textitФрагментальнаСхема()
    ІніціалізуватиПараметриРозподілу()
    P = ЗгенеруватиПочатковуПопуляцію()
    s = ОптимальнийРозв'язок(P)
    foreach p є P:
        Вік[p] = 0
    while not ВиконуєтьсяУмоваЗавершення():
        foreach p є P:
            Вага[p] = ОбчислитиВагу(p)
        P' = ВідбірЗПовтореннями (P, Вік, Вага)
        if НеобхідноРозширювати():
            foreach p є P':
                if Довжина(p)<n:
                    if МожнаРозширити(p):
                        p = Розширити(p)
                    else:
                        p = ВипадковаЗгортка()
                        Вік[p] = 0
            ПоновитиПараметриРозподілу()
        foreach p є P':
            if ВдалаМутація(p, p'):
                p=p'
                Вік[p] = Вік[p]+1
            else:
                p = ВипадковаЗгортка()
                Вік[p] = 0
        P = P'
        s = ОптимальнийРозв'язок(P ∪ {s})
    return S
```

Рис 1. Схема алгоритму

Наступним кроком проводиться відбір нової популяції. Процедура відбору є стохастичною, ймовірність вибрати певний елемент залежить від його ваги (чим більша вага, тим більша ймовірність) і віку (чим менший вік, тим більша ймовірність). В запропонованій реалізації алгоритму ймовірність обрати елемент пропорційна значенню ваги — віку з додаванням одиниці.

Далі проводиться перевірка, чи доречно проводити розширення згорток, тобто збільшувати довжину елементів популяції. Найпростіший варіант —

проводити розширення кожні k ітерацій (k — параметр алгоритму). Адаптивна реалізація може враховувати середнє значення енергії і вік популяції на заданій ітерації.

Розширення проводиться в два кроки — добудова структур та зміна параметрів розподілу. Під добудовою структур в популяції мається на увазі додання по одному амінокислотному залишку в початок і кінець заданої згортки у випадковому напрямку (за умови, що $l < n$). Якщо допустимої згортки таким чином згенерувати не вдалося, то вона замінюється на випадкову відповідної довжини, її вік встановлюється нульовим.

Функція *ПоновитиПараметриРозподілу*() поновлює значення параметрів M, σ та l таким чином, щоб індекси елементів, що найбільш ймовірно будуть підлягати зміні в структурі, рухались від середини згортки до її країв в процесі роботи алгоритму. Значення параметру l збільшується відповідно до довжини елементів в популяції (на 2, якщо $l < n$, або не змінюється в іншому випадку). Значення M та σ теж збільшуються на фіксовані величини; в запропонованій реалізації алгоритму M інкрементується на 1, σ не змінюється.

Останнім кроком ітерації є мутація. Її схема наведена на рис. 2.

```
ВдалаМутація(p, p')
    while not ВиконуєтьсяУмоваЗавершенняМутації()
        p'=p
        i = ОбратиІндекс()
        foreach можливих напрямків d:
        p'[i]=d
        if E(p')<E(p):
            return 1
    return 0
```

Рис 2. Схема процедури мутації

Процедура мутації є ітеративною. Функція *ВиконуєтьсяУмоваЗавершенняМутації*() в найпростішій реалізації базується на кількості кроків, які було виконано. Функція *ОбратиІндекс*() повертає випадкове значення відповідно до введеного розподілу. Далі, елемент згортки на позиції, що задана обраним індексом, послідовно замінюється на кожен елемент кодування, і, якщо найменша енергія зі знайдених, нижча за енергію початкової структури, відповідна згортка повертається функцією, вважається, що процедура мутації пройшла успішно. Якщо ж таку структуру знайти не вдалось, повертається значення 0.

4. Результати та висновки

Запропонований алгоритм було реалізовано та апробовано на задачах з бібліотеки Protein Data Bank. Наступними кроками досліджень є налаштування параметрів алгоритму для досягнення найвищих показників ефективності і порівняння результатів прогнозування з результатами інших існуючих алгоритмів.

Література

1. Dill K. A. Theory for the folding and stability of globular proteins. // Biochemistry. — 1985. — №24(6). — P. 1501–1509.
2. Unger R, Moult J. Finding the lowest free energy conformation of a protein is an NP-hard problem: proof and implications. // Bulletin of Mathematical Biology. — 1993. — №55(6). — P. 1183–1198.
3. Fidanova S., Lirkov I. Ant Colony System Approach for Protein Folding. // Int. Conf. Multiconference on Computer Science and Information Technology. — 2008. — P. 887–891.
4. Гуляницкий Л. Ф., Рудык В. А. Проблема предсказания структуры протеина: формализация с использованием кватернионов. // Кибернетика и системный анализ. — 2013. —№4. — С.130–137.
5. Shehu, A., De Jong, K. Evolutionary Algorithms for Protein Structure Modeling. // Proceedings of the Companion Publication of the 2015 on Genetic and Evolutionary Computation Conference. — 2015. P. 533–545.
6. Custodio F. L., Helio B., Laurent E. D. A multiple minima genetic algorithm for protein structure prediction. // Applied Soft Computing. — 2014. — №15. — P. 88–99.
7. Abu D. U., Steinhofel K. A hybrid approach to protein folding problem integrating constraint programming with local search. // BMC bioinformatics. — 2010. —№11(1). — S39.
8. Frausto-Solis J. Threshold temperature tuning Simulated Annealing for Protein Folding Problem in small peptides. // Computational and Applied Mathematics. — 2013. — №32(3). — P. 471–482.

Інститут кібернетики імені В. М. Глушкова НАН України, пр. Академіка Глушкова, 40, Київ, 03680 МСП, Україна.

Надійшла 09.11.2014