Xây dựng chương trình tối ưu hóa gene dựa trên thuật giải tối ưu đàn kiến cho *Escherichia coli*

Võ Trí Nam, Lê Đăng Lộc, Huỳnh Quốc Việt, Trần Linh Thước, Nguyễn Đức Hoàng

***Tóm tắt*—Trong sản xuất protein tái tổ hợp, việc chuyển một gene hoang dại của sinh vật này vào một chủng chủ biểu hiện khác đôi khi cho kết quả biểu hiện thấp vì sự không tương thích giữa gene và hệ thống biểu hiện. Trong trường hợp đó, gene mục tiêu cần được tối ưu hóa để tương thích hơn với hệ thống biểu hiện thông qua việc thay đổi thành phần nucleotide của nó bằng những codon đồng nghĩa (synonymous codons) nhưng vẫn không làm thay đổi trình tự protein. Trong các chương trình tối ưu hóa gene hiện nay, nhiều thuật giải tìm kiếm đã được áp dụng như Thuật Giải Di Truyền (Genetic Algorithm) hay Cửa Sổ Trượt (Sliding Window) nhằm tìm ra trình tự tối ưu. Trong nghiên cứu này, chúng tôi sử dụng thuật giải Tối Ưu Đàn Kiến (Ant Colony Optimization) để xây dựng chương trình tối ưu hóa gene. Kết quả cho thấy gene sau khi tối ưu hóa đã cải thiện xu hướng sử dụng codon (codon uasage), thành phần GC và hạn chế các yếu tố cản trở quá trình phiên mã, dịch mã như polycodon, polynucleotide, trình tự lặp lại và trình tự Shine - Dalgarno. Kết quả so sánh với các chương trình hiện nay cho gene mã hóa cho insulin người cũng cho thấy hiệu quả của chương trình xây dựng từ nghiên cứu. Những kết quả này đã chứng tỏ tiềm năng ứng dụng của thuật giải Tối Ưu Đàn Kiến cho vấn đề tối ưu hóa gene.**

***Từ khóa*—Thiết kế gene, tối ưu đàn kiến, tối ưu hóa codon, tối ưu hóa gene**

1 GIỚI THIỆU

Tối ưu hóa gene là một kỹ thuật nhằm gia tăng mức độ biểu hiện gene thông qua việc tăng cường hiệu quả phiên mã và dịch mã. Điều này được thực hiện bằng cách cải thiện các tiêu chí ảnh hưởng đến biểu hiện gene như xu hướng sử dụng codon, thành phần GC hay trình tự lặp lại. Nguyên tắc của tối ưu hóa gene là những codon đồng nghĩa có thể thay thế lẫn nhau để làm thay đổi thành phần nucleotide và từ đó thay đổi đặc tính của gene mà vẫn giữ nguyên trình tự amino acid tạo ra.

Vấn đề của tối ưu hóa gene là số lượng trình tự gene có thể mã hóa cho cùng một chuỗi polypeptide là rất lớn. Việc khảo sát tất cả các trình tự sẽ tốn kém rất nhiều thời gian và tài nguyên máy tính. Do đó, nhiều nghiên cứu đã đưa ra các thuật giải tìm kiếm trình tự gene tối ưu hóa. Trong số đó, Thuật Giải Di Truyền (Genetic Algorithm), thông qua việc mô phỏng lại sự tiến hóa của sinh vật trong môi trường sống dưới áp lực của chọn lọc tự nhiên, đã được áp dụng vào một số chương trình tối ưu hóa gene như Visual Gene Developer hay DNA 2.0 [4]. Tuy nhiên, do thuật giải này dựa theo nguyên lý của lọc tự nhiên luôn loại bỏ những trình tự xấu nên có thể dẫn đến kết quả tối ưu cục bộ. Năm 2010, Raab và cộng sự [9] đã đề xuất chương trình GeneOptimizer sử dụng thuật giải Cửa Sổ Trượt (Sliding Window). Thuật giải này tối ưu hóa các trình tự con (sub-sequence) chồng lắp lên nhau bằng một "cửa sổ biến thiên" dịch chuyển từ đầu đến cuối trình tự acid amin. Tuy nhiên điều đó đã làm giảm đi tính linh động của thuật giải này khi đánh giá và lựa chọn các trình tự do sự dịch chuyển "một chiều" của "cửa sổ biến thiên", dẫn đến kết quả tối ưu không phải lúc nào cũng đạt được. Thuật giải Tối Ưu Đàn Kiến (**A**nt **C**olony **O**ptimization-ACO), được giới thiệu bởi Dorigo vào năm 1992, là một thuật giải mạnh và đang được ứng dụng trong nhiều lĩnh vực khác nhau [2, 7]. ACO mô phỏng lại

Ngày nhận bản thảo: 09-7-2017; Ngày chấp nhận đăng: 23-7-2018; Ngày đăng: 30-8-2018

Võ Trí Nam[1,2], Lê Đăng Lộc[1], Huỳnh Quốc Việt[1], Trần Linh Thước[1], Nguyễn Đức Hoàng[1,3,*] – *[1]Trung tâm Khoa học và Công nghệ sinh học, [2]Phòng Thí nghiệm Công nghệ sinh học phân tử, [3]Bộ môn Vi sinh, Trường Đại học Khoa học Tự nhiên, ĐHQG-HCM*

*Email: ndhoang@hcmus.edu.vn

hành vi của những con kiến trong tự nhiên trong lúc chúng di chuyển qua những con đường có chiều dài khác nhau dẫn tới nguồn thức ăn. Cụ thể, mỗi con kiến sẽ tiết ra một loại hóa chất gọi là pheromone trên những con đường mà nó đi qua. Chất này sẽ kích thích các con kiến khác lựa chọn đi các con đường đó lần nữa. Sau một khoảng thời gian đủ lớn, bởi vì con đường ngắn nhất sẽ tích lũy nồng độ pheromone cao nhất nên phần lớn kiến sẽ hội tụ về con đường này. Bằng cách điều chỉnh nồng độ pheromone, ACO giúp làm giảm nguy cơ tối ưu cục bộ. Bên cạnh đó, thuật giải này ngoài việc tính điểm cho toàn bộ con đường (tương ứng với trình tự gene trong bài toán tối ưu hóa gene) thì còn thể hiện được độ tối ưu trên từng đoạn nhỏ; và việc chọn đường đi được thực hiện lặp lại nhiều lần theo thời gian nên đảm bảo được tính linh động trong quá trình tìm kiếm kết quả. Chúng tôi đã xây dựng chương trình tối ưu hóa gene dựa trên thuật giải ACO, sau đó thực hiện đánh giá và so sánh với các chương trình khác nhằm kiểm tra khả năng ứng dụng của thuật giải ACO trong vấn đề tối ưu hóa gene.

2 VẬT LIỆU VÀ PHƯƠNG PHÁP

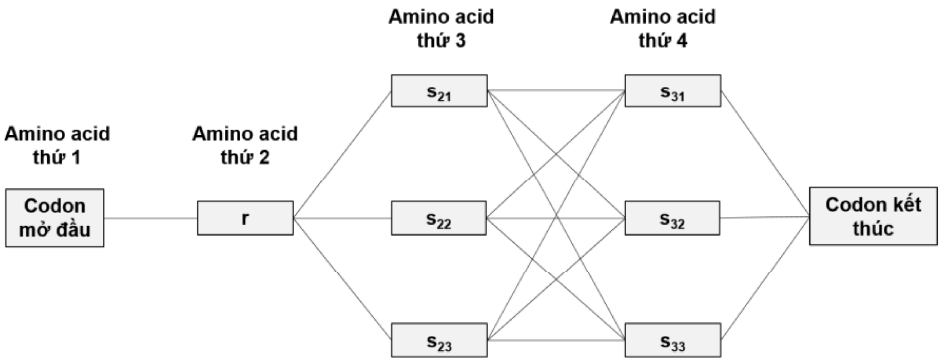
Thu nhận và phân tích dữ liệu trình tự

Dữ liệu nhóm gene biểu hiện cao (**H**ighly **E**xpressed **G**enes - HEG) của *Escherichia coli* str. K12 substr. W3110 được thu nhận từ cơ sở dữ liệu HEG-DB tại địa chỉ genomes.urv.cat/HEG-DB. Từ đó, độ thích nghi tương đối của codon (w_c) được tính toán dựa theo nghiên cứu của Cannarozzi và cộng sự [1]. Trình tự gene mã hóa insulin của người được thu nhận từ cơ sở dữ liệu NCBI với mã số accession NC_000011.10 có chiều dài 333 nucleotide được sử dụng để đánh giá chương trình.

Xây dựng chương trình tối ưu hóa gene

Mô hình bài toán

Thông qua việc mô phỏng những con đường đi bằng các trình tự nucleotide mã hóa cho chuỗi polypeptide, mô hình bài toán sẽ có dạng như Hình 1. Tổ kiến và nguồn thức ăn lần lượt là codon mở đầu và codon kết thúc. r là codon hiện tại mà một con kiến (kiến nhân tạo) đang ở đó. s là những codon cùng mã hóa cho các amino acid kế tiếp tương ứng với các con đường mà con kiến có thể chọn. Chiều dài của một đường đi là điểm số ($Score_{Total}$ – được nêu trong phần hàm tính điểm) của trình tự nucleotide tương ứng.



Hình 1. Thí dụ về mô hình của bài toán tối ưu hóa gene bằng thuật giải ACO

Thuật giải tối ưu đàn kiến

Hàm tính điểm

Hàm tính điểm tổng:

Tổng điểm của mỗi trình tự nucletiode được tính bằng công thức sau: [9]

$$Score_{Total} = \sum_{q=1}^{\text{Số lượng tiêu chí tối ưu hóa}} G_q Score_q$$

Trong đó: G_q: Trọng số của mỗi tiêu chí tối ưu hóa. Chương trình sẽ mặc định là 1 (Người dùng có thể tùy chọn)

$Score_q$: Lần lượt là $Score_{CAI}$, $Score_{GC}$, $Score_{PC}$, $Score_{PN}$, $Score_{RS}$, $Score_{SD}$, $Score_{HSC}$. Mỗi Score thành phần này là độc lập và được như sau:

Xu hướng sử dụng codon:

Số điểm ($Score_{CAI}$) được tính dựa trên chỉ số CAI như sau: [1]

$$Score_{CAI} = exp(\frac{1}{O_{tot}} \sum_{c \in C} \log w_c)$$

Trong đó: O_{tot}: Tổng số codon của gene cần tối ưu hóa

w_c: Độ thích nghi tương đối của một codon của tập HEG

Thành phần GC:

$$\text{ScoreGC} = \begin{cases} GC_{min} - GC \ (GC < GC_{min}) \\ GC - GC_{max} \ (GC > GC_{max}) \\ 0 \ (GC_{min} \leq GC \leq GC_{max}) \end{cases}$$

Trong đó: GC: %GC của gene cần tối hóa

GC_{min} = 0,5056 (*)

GC_{max}= 0,5367 (*)

(*) Được xác định thông qua tứ phân vị Q1 (25%) và Q3 (75%) trong tập HEG bằng đồ thị boxplot.

Polycodon:

$$Score_{PC_i} = \sum_{i=1}^{n} \frac{len(PC_i) - const}{const}$$

Trong đó: $len(PC_i)$: Chiều dài polycodon thứ i

n: Số polycodon có chiều dài $\geq const$ với $const = 3$ (**)

Polynucleotide:

$$Score_{PN_i} = \sum_{i=1}^{n} \frac{len(PN_i) - const}{const}$$

Trong đó: $len(PN_i)$: Chiều dài polynucleotide thứ i

n: Số polynucleotide có chiều dài ≥ const với $const = 5$ (**)

Trình tự lặp lại:

$$Score_{RS_i} = \sum_{i=1}^{n} \frac{len(RS_i) - const + 1}{d - len(RS_i)}$$

Trong đó: $len(RS_i)$: Chiều dài trình tự lặp lại thứ i

d: Khoảng cách giữa 2 trình tự lặp lại

n: Số cặp trình tự lặp lại có chiều dài ≥ const với const = 7 (**)

Trình tự Shine - Dalgarno:

$$Score_{SD} = \sum_{i=1}^{n} \frac{len(SD) - mismacth_i}{len(SD)}$$

Trong đó: $len(SD)$: Chiều dài trình tự Shine - Dalgarno

$mismacth_i$: Số nucleotide sai khác của trình tự thứ i so với trình tự Shine – Dalgarno bảo tồn

n: Số trình tự tương đồng với trình tự Shine – Dalgarno bảo tồn với $mismacth = \{0; 1\}$ (**)

Mã kết thúc ẩn:

$$Score_{HSC} = \min\left(\frac{HSC_2}{5}, 1\right) + \min\left(\frac{HSC_3}{5}, 1\right)$$

Trong đó: HSC_2: Số lượng mã kết thúc ẩn nằm ở khung đọc 2

HSC_3: Số lượng mã kết thúc ẩn nằm ở khung đọc 3

(**) Các thông số $const$ và $mismacth$ trong các công thức trên được tham khảo từ các giá trị khảo sát trong nhóm nghiên cứu Tin - Sinh học Trường ĐH Khoa học Tự nhiên, ĐHQG-HCM [5].

Những công thức trong các bước tiếp theo đều dựa trên nghiên cứu về thuật giải đàn kiến của Dorigo và Gambardella về bài toán người bán hàng (Traveling Salesman Problem) [3]. Tuy nhiên, chúng sẽ có vài sự biến đổi để phù hợp với bài toán tối ưu hóa gene.

Xác định nồng độ pheromone ban đầu

Sự di chuyển của những con kiến phụ thuộc vào nồng độ pheromone, vì thế để khởi động bài toán, tất cả các con đường sẽ được đặt một giá trị nồng độ pheromone ban đầu τ0 theo công thức:

τ0 = $Score_{C_{nn}}$/n

Trong đó: n: Chiều dài của chuỗi polypeptide

$Score_{c_{nn}}$: Điểm số của trình tự tối ưu được tìm bằng thuật giải Nearest Neighbor [10]. Đặt một con kiến ở vị trí codon khởi đầu. Khi đó, codon khởi đầu là vị trí hiện tại của con kiến nên gọi là r. Còn s là những codon cùng mã hóa cho các amino acid kế tiếp. Sau đó, chúng ta tìm một codon s sao cho $Score_{Total}\ (r, s)$ là lớn nhất rồi đặt con kiến ở codon s đó. Như vậy, codon s này sẽ trở thành một codon r mới. Tiếp tục như thế sẽ tìm một codon s mới cho tới khi kết thúc chiều dài trình tự gene.

Xây dựng đường đi

Một con kiến tại codon r lựa chọn một codon s để di chuyển dựa theo quy luật được cho bởi công thức sau:

$$p=\begin{cases}\text{argmax}\{[\tau(r,s)]\ .\ [\eta(r,s)]^{1/\beta}\} & \text{với } s\in J_k(r) \text{ nếu } q\le q_0\\ S & \text{nếu } q>q_0\end{cases}$$

Trong đó: $arg\ max$: arguments of the maxima, nghĩa là tìm codon s sao cho $[\tau(r,s)]\ .\ [\eta(r,s)]^{1/\beta}$ đạt giá trị lớn nhất

$\tau(r,s)$: Nồng độ pheromone trên cạnh (r,s)

$\eta(r,s)$: Điểm số trình tự tính từ codon khởi đầu tới codon s

β: Hệ số ảnh hưởng của η được khảo sát bên dưới

q: Giá trị phát sinh ngẫu nhiên trong đoạn [0; 1]

q_0: Giá trị cố định trong đoạn [0; 1] được khảo sát bên dưới

$J_k(r)$: Tập hợp các codon s mà con kiến thứ k tại codon r có thể lựa chọn

S: Codon ngẫu nhiên được lựa chọn dựa theo phân phối xác suất được cho bởi công thức:

$$p_k(r,s)=\frac{[\tau(r,s)]\ .\ [\eta(r,s)]^{1/\beta}}{\sum \tau[r,z]\ .\ [\eta(r,z)]^{1/\beta}}$$

Trong đó: $s, z \in J_k(r)$

Cập nhật pheromone cục bộ

Sau khi mỗi con kiến di chuyển từ codon r tới codon s, nồng độ pheromone trên cạnh (r,s) sẽ được cập nhật bằng công thức sau:

$$\tau(r,s)=(1-\rho)\ .\tau(r,s)+\ \rho\ .\tau_0$$

Trong đó: ρ: Tỉ lệ pheromone bay hơi được khảo sát bên dưới

τ_0: Nồng độ pheromone ban đầu

Sau khi tất cả các con kiến trong mỗi vòng lặp đã hoàn thành xong đường đi của chúng từ codon khởi đầu tới codon kết thúc, các cập nhật pheromone cục bộ sẽ được phục hồi lại để trả về trạng thái đầu vòng lặp, sau đó tiến hành cập nhật pheromone toàn cục.

Cập nhật pheromone toàn cục

Quá trình cập nhật pheromone toàn cục chỉ áp dụng trên những cạnh thuộc trình tự có điểm số cao nhất. Quá trình cập nhật được thực hiện theo công thức sau:

$$\tau(r,s)=(1-\rho)\ .\tau(r,s)+\ \rho\ .\Delta\tau_{(r,s)}^{best}$$

Trong đó: ρ: Tỉ lệ pheromone bay hơi được khảo sát bên dưới

$\Delta\tau_{(r,s)}^{best}$: Nồng độ pheromone thêm vào được tính theo công thức:

$$\Delta\tau_{(r,s)}^{best}=L_{best}$$

Trong đó L_{best} là điểm số cao nhất của trình tự gene hoàn chỉnh tính cho tới thời điểm hiện tại

Điều kiện kết thúc

Quá trình tối ưu hóa lặp lại từ bước Xây dựng đường đi cho tới khi số vòng lặp đạt tới một giá trị cho trước, bài toàn sẽ kết thúc và đưa ra trình tự gene tối ưu.

Khảo sát tìm bộ thông số tối ưu cho chương trình tối ưu hóa gene

Những thông số của thuật giải ACO được khảo sát bao gồm β = {2; 3; 4; 5}, ρ = {0,1; 0,3; 0,5}, q0 = {0,5; 0,7; 0,9}, số lượng kiến là 10 con và số vòng lặp từ 1 tới 150 vòng.

Đánh giá chương trình tối ưu hóa gene

Bộ thông số tối ưu sau khi khảo sát sẽ được sử dụng để tối ưu hóa trình tự gene mã hóa insulin để đánh giá hiệu quả của chương trình. Gene trước và sau khi tối ưu hóa được so sánh dựa trên các yếu tố ảnh hưởng đến khả năng biểu hiện gene

bao gồm: Xu hướng sử dụng codon, thành phần GC, polycodon, polynucleotide, trình tự lặp lại, trình tự Shine - Dalgarno, mã kết thúc ẩn và khả năng loại bỏ trình tự nhận biết của enzyme cắt giới hạn. Trong bài báo này, chương trình được đánh giá trên máy tính Linux OS, Intel Core i3, 3.5 GHz, với 8 GB RAM.

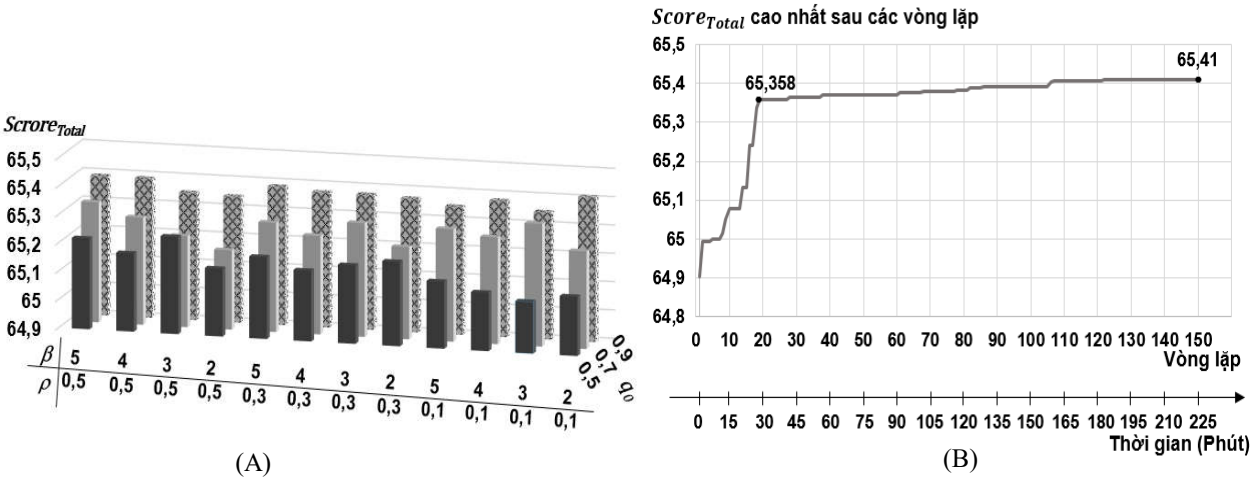
So sánh với các chương trình tối ưu hóa gene khác

Các chương trình dùng để so sánh bao gồm: OPTIMIZER tại địa chỉ genomes.urv.es/OPTIMIZER, Codon Optimization OnLine – COOL tại địa chỉ omictools.com/codon-optimization-online-tool, và GeneOptimizer [9].

3 KẾT QUẢ VÀ THẢO LUẬN

Xây dựng chương trình tối ưu hóa gene và khảo sát bộ thông số tối ưu

Chúng tôi đã thu nhận được 255 gene HEG ở *E. coli* K12 từ HEG-DB và gene mã hóa insulin người từ NCBI. Dữ liệu HEG sau đó được dùng để tính toán các giá trị wc để sử dụng trong chương trình tối ưu hóa gene. Sau khi tạo được chương trình tối ưu hóa gene, gene mã hóa insulin người được sử dụng để minh họa. Gene mã hóa insulin trước tối ưu hóa (trình tự thu nhận từ NCBI) đã được tính toán $Score_{Total}$=59,768 dựa theo hàm tính điểm tổng. Sau đó, sự khảo sát các thông số β, ρ, q0 được thực hiện với số vòng lặp được cố định là 150. Kết quả được thể hiện trên Hình 2A.



(A) (B)

Hình 2. Kết quả khảo sát các thông số.
(A) Các thông số cho chương trình tối ưu hóa gene, (B) Bộ thông số (β; ρ; q_0) = (2; 0,1; 0,9)

Theo đó $Score_{Total}$ của gene mã hóa insulin sau tối ưu hóa luôn đạt kết quả cao hơn so với trước tối ưu hóa. Bên cạnh đó, tất cả các tổ hợp của các thông số ứng với giá trị q0 càng lớn thì cho điểm số càng cao trừ một trường hợp của tổ hợp (β; ρ; q0) = (3; 0,5; 0,7) lại cho kết quả thấp hơn tổ hợp (β; ρ; q0) = (3; 0,5; 0,5). Tuy nhiên, khi q0 đạt giá trị cực đại tức q0 =1, thuật giải ACO sẽ cho kết quả giống như thuật giải Nearest Neighbor và tất cả các tổ hợp (β; ρ; q0) đều cho cùng một kết quả khá thấp $Score_{Total}$ = 64,909 (kết quả không hiển thị trên biểu đồ). Vì thế, q0 càng lớn thường cho kết quả càng cao nhưng nên nhỏ hơn 1.

Bộ thông số (β; ρ; q0) = (2; 0,1; 0,9) cho điểm số cao nhất. Vì thế, bộ thông số này sẽ được chọn là bộ thông số tối ưu cho chương trình tối ưu hóa gene. Điểm số của trình tự sau khi tối ưu hóa tính theo số vòng lặp (từ 1 đến 150 vòng) và thời gian của bộ thông số này thể hiện trên Hình 2B. Theo đó, khi số vòng lặp càng tăng thì điểm số cao nhất càng tăng nhưng biên độ tăng chậm dần. Cụ thể, điểm số tăng nhanh trong 19 vòng lặp đầu tiên (tăng 0,46 điểm), sau đó tăng rất chậm, từ vòng lặp số 19 đến 150 chỉ tăng được khoảng 0,05 điểm. Điều này chứng tỏ sau 19 vòng lặp, nồng độ pheromone ở các con đường có chiều dài ngắn (số điểm cao) đã tăng cao hơn các con đường còn lại nên có sự tập trung của đàn kiến ở các con đường này. Từ vòng lặp 20 trở đi, chương trình tiếp tục

cải thiện kết quả tối ưu nhưng biên độ chậm hơn do độ tối ưu của lời giải đã đạt độ cao nhất định, hay nói cách khác là bắt đầu có sự hội tụ. Từ vòng lặp số 107, điểm số dường như không tăng nữa (tăng rất ít). Vì vậy, đối với bộ thông số này, để có kết quả tốt nhất, nghiên cứu khuyến cáo nên chạy từ 110 vòng lặp trở lên. Thời gian xử lý của chương trình trong trường hợp này khoảng 165 phút (với gene insulin dài 333 nucleotide).

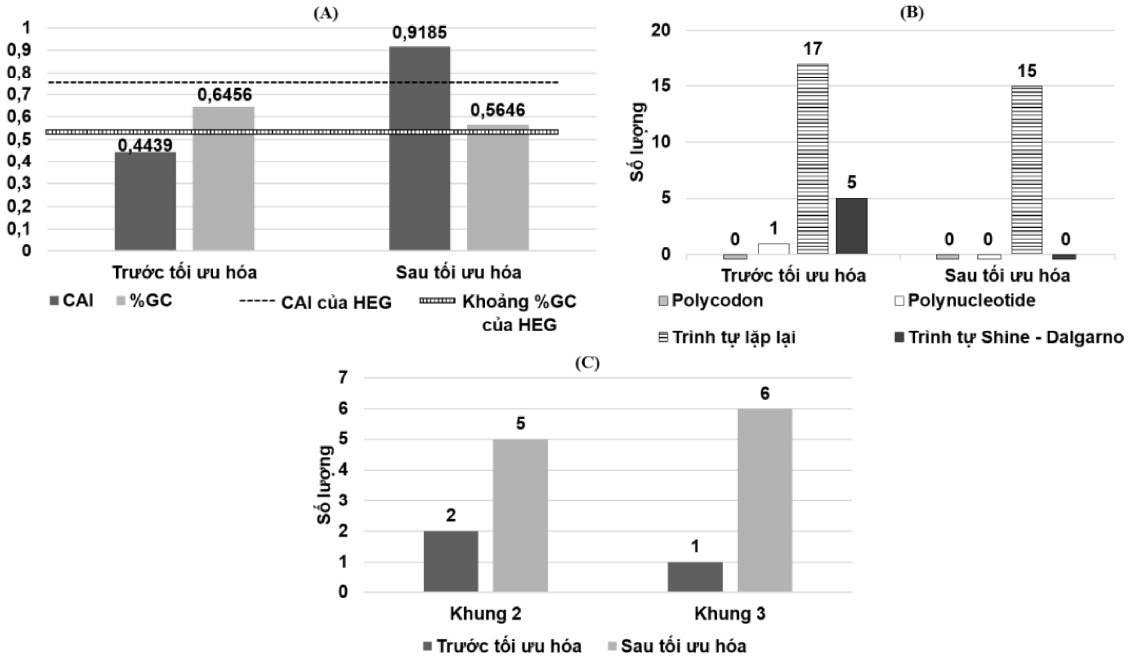
Đánh giá chương trình tối ưu hóa gene

Xu hướng sử dụng codon

Xu hướng sử dụng codon là tiêu chí được sử dụng trong tất cả các chương trình tối ưu hóa gene hiện nay bởi tầm quan trọng tiêu chí này trong biểu hiện gene [6]. Xu hướng sử dụng codon biểu thị qua chỉ số CAI, có giá trị từ 0 đến 1, thể hiện tổng thể khả năng phù hợp giữa thành phần codon trong gene với xu hướng sử dụng codon của hệ thống biểu hiện. Hình 3A cho thấy CAI của gene sau tối ưu hóa cao hơn 2 lần so với trước tối ưu hóa và cao hơn giá trị trung bình của nhóm HEG (0,75) 1,2 lần. Như vậy, chương trình đã thành công trong việc cải thiện CAI của gene ban đầu.

Thành phần GC

Thành phần GC (%GC) ảnh hưởng đến độ mạnh liên kết giữa 2 mạch của phân tử DNA nên nó tác động lớn đến mức độ biểu hiện gene [9]. So với trước tối ưu hóa, %GC của gene sau tối ưu hóa đã được điều chỉnh về gần ngưỡng %GC của HEG (0,5056 đến 0,5367) (Hình 3A). Điều đó cho thấy hiệu quả tối ưu hóa %GC của chương trình.



Hình 3. Kết quả đánh giá chương trình tối ưu hóa gene.
(A) Xu hướng sử dụng codon, thành phần GC.
(B) Polycodon, polynucleotide, trình tự lặp lại, và trình tự Shine – Dalgarno. (C) Mã kết thúc ẩn

DNA motif

Các DNA motif làm giảm hiệu quả biểu hiện gene như polycodon, polynucleotide, trình tự lặp lại và trình tự Shine - Dalgarno đều đã được chương trình loại bỏ hay giảm số lần xuất hiện. Cụ thể, số lượng polycodon, polynucleotide và trình tự Shine - Dalgarno sau tối ưu hóa đã được chương trình giảm về 0 và trình tự lặp lại được giảm đi 2 trình tự (Hình 3B). Như vậy chương trình đã giảm được các yếu tố tiêu cực ảnh hưởng đến biểu hiện gene.

Mã kết thúc ẩn

Mã kết thúc ẩn là những codon kết thúc nằm trong vùng mã hóa của gene nhưng thuộc khung đọc 2 hoặc khung đọc 3. Mã kết thúc ẩn có vai trò giúp dừng sự dịch mã lệch khung nhằm tránh tạo ra các sản phẩm protein sai và tiết kiệm thời gian cũng như vật chất bên trong tế bào [8]. Gene sau tối ưu hóa đã có sự gia tăng số lượng mã kết thúc ẩn ở cả 2 khung đọc (Hình 3C). Như vậy chương trình đã thành công trong việc tăng cường sự xuất hiện của các mã kết thúc ẩn trong gene.

Trình tự nhận biết của enzyme cắt giới hạn

Cuối cùng, chương trình được đánh giá dựa trên khả năng loại bỏ vùng nhận biết của các enzyme cắt giới hạn. Thí dụ, có 3 trình tự nhận biết của enzyme $P_{st}I$ trên gene mã hóa insulin có thể được để loại bỏ trong trình tự tối ưu. Vị trí nhận biết của enzyme này trên trình tự tối ưu có thể được thay đổi cho phù hợp (kết quả không được thể hiện). Như vậy, chương trình tối ưu hóa gene đã thành công trong việc tối ưu các tiêu chí xu hướng sử dụng codon, thành phần GC, các DNA motif và trình tự nhận biết của enzyme cắt giới hạn. Tiếp theo, tiến hành so sánh hiệu quả tối ưu hóa của chương trình xây dựng được với các chương trình tối ưu hóa gene khác hiện nay để có kết luận khách quan hơn về khả năng tối của chương trình.

So sánh với các chương trình tối ưu hóa gene hiện nay

Các chương trình được sử dụng để so sánh bao gồm OPTIMIZER, GeneOptimizer và COOL. Số lượng và các loại tiêu chí tối ưu hóa gene trong các chương trình là khác nhau. Vì thế để có sự khách quan, nghiên cứu chỉ chọn ra các tiêu chí giống nhau giữa các chương trình để thực hiện so sánh.

Kết quả so sánh với hai chương trình OPTIMIZER và GeneOptimizer được thể hiện trên Bảng 1. Theo đó chương trình xây dựng cải thiện chỉ số CAI cao hơn 2 chương trình còn lại. Về %GC, chương trình xây dựng cho kết quả tốt hơn OPTIMIZER (độ lệch thấp hơn 2,6 lần) nhưng vẫn chưa nằm trong khoảng %GC của HEG như GeneOptimizer. Tương tự, kết quả so sánh giữa chương trình xây dựng với chương trình COOL ở Bảng 2 cũng cho thấy sự cải thiện CAI của chương trình xây dựng tốt hơn so với COOL. Cụ thể, chương trình xây dựng cải thiện được chỉ số CAI tăng lên 0,45 điểm, trong khi đó giá trị này là 0,23 điểm đối với COOL. Ngược lại, các tiêu chí như %GC, trình tự lặp lại, mã kết thúc ẩn thì COOL cho hiệu quả cải thiện tốt hơn.

Dựa vào các kết quả trên, chương trình xây dựng từ nghiên cứu đã cho thấy khả năng tối ưu hóa tương đương so với các chương trình khác. Tuy nhiên, mỗi chương trình cho hiệu quả tối ưu ở mỗi tiêu chí khác nhau. Chương trình xây dựng từ nghiên cứu vượt trội ở xu hướng sử dụng codon so với cả 3 chương trình được so sánh, ngược lại cho hiệu quả kém hơn ở các tiêu chí như %GC hay trình tự lặp lại. Mức độ tối ưu hóa ở các tiêu chí phụ thuộc vào hàm tính điểm tổng. Do đó, với sự thay đổi trọng số trong hàm tính điểm tổng, chương trình được dự đoán có thể cho phép người dùng chọn lựa mức độ tối ưu ở các tiêu chí cho phù hợp với nhu cầu sử dụng trong từng trường hợp cụ thể. Một thế mạnh khác của chương trình đã xây dựng là khả năng kết hợp đa dạng nhiều tiêu chí tối ưu hóa khác nhau cho phép người dùng lựa chọn so với 3 chương trình đã so sánh.

Bảng 1. Kết quả so sánh giữa chương trình xây dựng với GeneOptimizer và OPTIMIZER

Các tiêu chí tối ưu hóa gene	**Trước khi tối ưu hóa**	**Sau khi tối ưu hóa**		
		Chương trình xây dựng	GeneOptimizer	OPTIMIZER
CAI	0,4439	0,9852	0,8195	0,8090
%GC	0,6456	0,5646	0,5225	0,6096
Độ lệch so với khoảng %GC của HEG [0,5056; 0,5367]	0,1089	0,0279	Thuộc khoảng %GC HEG	0,0729

Bảng 2. Kết quả so sánh giữa chương trình xây dựng với COOL

Các tiêu chí tối ưu hóa gene		Trước khi tối ưu hóa	Sau khi tối ưu hóa	
			Chương trình xây dựng	COOL
CAI		0,4439	0,8937	0,6757
%GC		0,6456	0,5526	0,5285
Độ lệch so với khoảng %GC của HEG [0,5056; 0,5367]		0,1089	0,0159	Thuộc khoảng %GC HEG
Trình tự lặp lại		17	15	0
Polynucleotide		1	0	0
Mã kết thúc ẩn	Khung đọc 2	2	5	5
	Khung đọc 3	1	5	9
Trình tự Shine - Dalgarno		5	0	0

4 KẾT LUẬN

Nghiên cứu đã xây dựng thành công chương trình tối ưu hóa gene dựa trên thuật giải Tối Ưu Đàn Kiến. Bộ thông số tối ưu cho chương trình là $\beta = 2$, $\rho = 0,1$, $q0 = 0,9$ với số vòng lặp từ 110 trở lên. Chương trình xây dựng cho khả năng tối ưu hóa tương đương với các chương trình Gene Optimizer, OPTIMIZER và COOL. Các kết quả trên thể hiện khả năng ứng dụng hiệu quả của thuật giải ACO cho vấn đề tối ưu hóa gene. Ngoài ra, để kết quả đánh giá chương trình được toàn diện và khách quan hơn, cần tiến hành tối ưu hóa bằng các trình tự gene khác ngoài insulin cũng như cần tiến hành thực nghiệm để kiểm tra khả năng biểu hiện gene sau tối ưu hóa.

Lời cảm ơn: *Nghiên cứu này được tài trợ bởi Trung tâm Khoa học và Công nghệ Sinh học và từ đề tài cấp Trường mã số T2016-22, Trường Đại học Khoa học Tự nhiên, ĐHQG-TP.HCM. Cảm ơn PGS.TS. Trần Văn Lăng đã góp ý sử dụng thuật giải Tối Ưu Đàn Kiến để giải quyết vấn đề.*

TÀI LIỆU THAM KHẢO

[1] G.M. Cannarozzi, A. Schneider, Codon evolution: mechanisms and models, Oxford University Press, 2012.

[2] M. Dorigo, "Optimization, learning and natural algorithms", Ph. D. Thesis, Politecnico di Milano, Italy, 1992.

[3] M. Dorigo, L.M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem", *IEEE Transactions on evolutionary computation*, vol. 1, pp. 53–66, 1997.

[4] J.H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.*, U Michigan Press, 1975.

[5] H.T. Le, N.T. Vo, T. Huynh, H.D. Nguyen, "Analyzing characteristics of Highly Expressed Genes (HEG) of *Escherichia coli* K12", *Genomic Medicine*, 2015.

[6] S. Lee, S. Weon, S. Lee, C. Kang, "Relative codon adaptation index, a sensitive measure of codon usage bias", *Evolutionary Bioinformatics*, vol. 6, no. 47, 2010.

[7] R.B. Pandhi, VLSI Circuit Partitioning using Ants Colony Optimization, PhD dissertation, Computer Science & Engineering Department, Thapar University, India, 2007.

[8] V. Phan, S. Saha, A. Pandey, T.Y. Wong, "Synthetic gene design with a large number of hidden stops", *International Journal of Data Mining and Bioinformatics*, vol. 4, pp. 377–394, 2010.

[9] D. Raab, M. Graf, F. Notka, T. Schödl, R. Wagner, "The GeneOptimizer Algorithm: using a sliding window approach to cope with the vast sequence space in multiparameter DNA sequence optimization", *Systems and Synthetic Biology*, vol. 4, pp. 215–225, 2010.

[10] D.J. Rosenkrantz, R.E. Stearns, P.M. Lewis, "An analysis of several heuristics for the traveling salesman problem", *SIAM Journal on Computing*, vol. 6, pp. 563–581, 1977.

Constructing a gene optimization program based on the ant colony optimization algorithm for *Escherichia coli*

Tri Nam Vo[1,2], Dang Loc Le[1], Quoc Viet Huynh[1], Linh Thuoc Tran[1], Duc Hoang Nguyen[2,3,*]

[1]Center for Bioscience and Biotechnology, [2]Laboratory of Molecular Biotechnology,
[3]Department of Microbiology, University of Science, VNUHCM

[*]Corresponding author: ndhoang@hcmus.edu.vn

Received: 09-7-2017; Accepted: 13-7-2018; Published: 30-8-2018

***Abstract*—In recombinant protein production, transferring a wild type gene of one organism into another expression host sometime resulted in a low gene expression due to incompatibility between the gene and the expression system. In that case, the target gene needed to be optimized to be more compatible with the expression system through gene optimization process in which nucleotide composition of original gene would be replaced by synonym codons while retaining the protein sequence. In existing gene optimization programs, many optimization algorithms have been applied, such as Genetic Algorithm or Sliding Window, to search for the optimized gene sequence. In this research, we applied the Ant Colony Optimization (ACO) algorithm to construct a gene optimization program. The results showed that the gene after optimization has been improved in codon usage, GC content and reduced the occurrence of factors reducing transcription and translation efficiencies such as polycodon, polynucleotide, repeated sequence, and Shine - Dalgarno sequence. Comparing with some current programs using a gene encoding for human insulin also proved the efficiency in the gene optimization this program. These results have demonstrated the capabilities of applying ACO algorithm in the gene optimization problem.**

***Keywords*—Ant Colony Optimization, Codon optimization, gene design, gene optimization**