用于桁架拓扑优化的统计萤火虫算法

Nghi Huu Duong[a]、Duy Vo[b, c]、Pruettha Nanakorn[a]

[a]土木工程与技术学院，诗琳通国际理工学院，法政大学，巴吞他尼，泰国

[b]维新计算工程研究院（*DTRICE*），维新大学，胡志明市，越南

[c]土木工程学院，维新大学，岘港，越南

摘要

本研究提出了一种用于桁架拓扑优化的算法，称为统计萤火虫算法（SFA）。在该算法中，利用萤火虫运动的历史结果进行假设检验，将萤火虫之间当前信息交换所建议的运动限制为可能有用的运动。假设检验被应用于普通萤火虫算法（FA）的机制，而不改变其结构。因此，所提出算法的实现简单直接。将萤火虫的运动限制在可能有用的范围内，可减少萤火虫评估次数，进而降低计算开销。为检验所提出算法的有效性和效率，本文使用该算法求解若干桁架拓扑优化问题，其中包括一些基准问题。结果表明，SFA中新增的统计策略在保持所得结果质量的同时，显著降低了计算开销，从而提升了原始FA的性能。

*Keyw*词：萤火虫算法、统计学、均值假设检验、桁架拓扑优化 n.

电子邮件地址： nghihd.ce@gmail.com（N. H. Duong）； voduy3@duytan.edu.vn（D. Vo）； nanakorn@siit.tu.ac.th（P. Nanakorn）

1. 引言

元启发式优化方法 [1, 2] 因其相较于传统优化方法的优势， [3, 4] 在结构优化领域越来越受到研究人员的青睐。尽管包括元启发式方法在内的优化方法在近几十年来得到了广泛研究，但仍有许多方面有待改进。这是因为现实中的结构

优化问题规模庞大且复杂，而大多数现有的最优算法只能处理规模较小、较简单的问题。差分进化（DE）[5-8], 、遗传算法（GAs）[9-11], 、粒子群优化（PSO）[12-15], 、萤火虫算法（FAs）[16-19], 以及基于教学学习的优化（TLBO） [20-23], 等元启发式方法，在寻找全局最优解方面较传统方法有所改进。元启发式方法利用多个搜索点进行搜索，因此搜索过程不易陷入局部最优区域。与传统方法不同，元启发式方法无需目标函数和约束函数的梯度，而这些梯度往往无法获取或计算成本高昂。在这些元启发式方法中，尽管萤火虫算法的应用规模不及其他算法，但已被证明是一种稳健可靠的方法 [24]。

桁架优化问题可分为三类，即尺寸、形状和拓扑优化 [25-27]。在尺寸优化中 [28-30], ，桁架杆件的截面积是设计变量，而节点坐标和连接关系保持不变。在形状优化中 [31-33], ， 节点坐标成为设计变量，而桁架杆件的截面积和连接关系保持不变。在拓扑优化中 [34-36], ，桁架杆件的连接关系是设计变量，通常使用桁架杆件的截面积来定义。因此，桁架拓扑优化本质上包含尺寸优化。在求解桁架拓扑优化问题时，必须定义包含所有可能桁架的搜索空间。地面结构法是定义桁架拓扑优化问题搜索空间最常用的方案。在该方法中，通过使用预定义节点之间的连接，将搜索空间定义为桁架单元的可能位置。可以仔细选择节点的位置和连接方式，以控制最终的搜索空间。地面结构法的一个显著缺点是，搜索空间的规模会随着节点数量的增加而迅速增大 [25, 37, 38]。当节点数量增加时，搜索维度的数量也会增加。具有大量搜索维度的优化问题往往也包含大量局部最优解，因此难以求解。这些问题不可避免地需要大量计算资源。

为避免陷入局部最优区域，包括萤火虫算法在内的稳健元启发式方法必须尽可能允许搜索点交换信息。这些信息交换所建议的操作通常需要大量计算资源。显然，只有执行有用的操作才是理想的。在萤火虫算法的机制中，每只萤火虫在移动过程中都必须与群体中的所有其他萤火虫通信。这些信息交换并不总能引导萤火虫前往更好的位置，而对移动的萤火虫进行评估则始终需要耗费大量时间。评估

结构优化中的设计解始终需要进行有限元分析（FEA），而这通常会耗费大量资源。因此，如果只执行那些有可能使萤火虫到达更佳位置的操作，将会更有益。

本研究提出了一种统计萤火虫算法（SFA）。该算法采用均值假设检验 [39-41]，利用过去信息交换的统计数据，预测当前信息交换所建议的即将采取的行动是否有用。通过使用假设检验，可以估计萤火虫对之间的信息交换所建议行动的有效性。对于萤火虫对所建议的、经假设检验判定其协作为无效的任何行动，将不予执行。对于采用地面结构法进行的大规模桁架拓扑优化问题，该策略尤其能够降低传统萤火虫算法的计算开销。

2. 假设检验

统计分析中的假设 [42, 43] 只是关于总体参数的一个主张或陈述，例如总体均值、方差和比例。在推断统计学中，研究的是样本，而不是整个总体。不过，所得结果会用于推断整个总体。本研究在所提出的萤火虫算法中使用总体均值的假设检验。在假设检验中，需要考虑两个相互补充的假设，即零假设和备择假设，后者是零假设的补集。进行假设检验，主要是为了决定接受还是拒绝零假设。如果零假设被拒绝，则足以支持备择假设。另一方面，如果零假设未被拒绝，则不足以支持备择假设。假设检验可分为两类，即双尾假设检验和单尾假设检验。令 μ 表示总体均值，μ_0 表示某个特定值。如图 1 所示，在双尾假设检验中，零假设（H_0）表明 $\mu = \mu_0$。因此，备择假设（H_a）表明 $\mu \neq \mu_0$。如图 2 所示，单尾假设检验包括两种情况，即左尾和右尾假设检验。在左尾检验中，零假设（H_0）表明 $\mu \geq \mu_0$，备择假设（H_a）则表明 $\mu < \mu_0$。在右尾检验中，零假设（H_0）表明 $\mu \leq \mu_0$，备择假设（H_a）则表明 $\mu > \mu_0$。

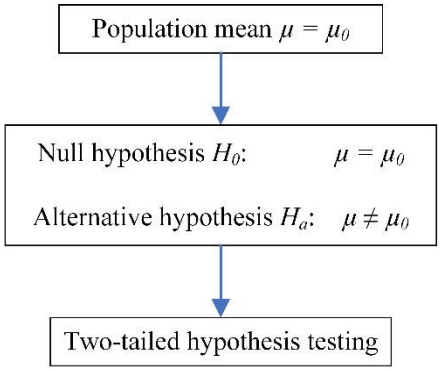

图 1. 均值的双尾假设检验

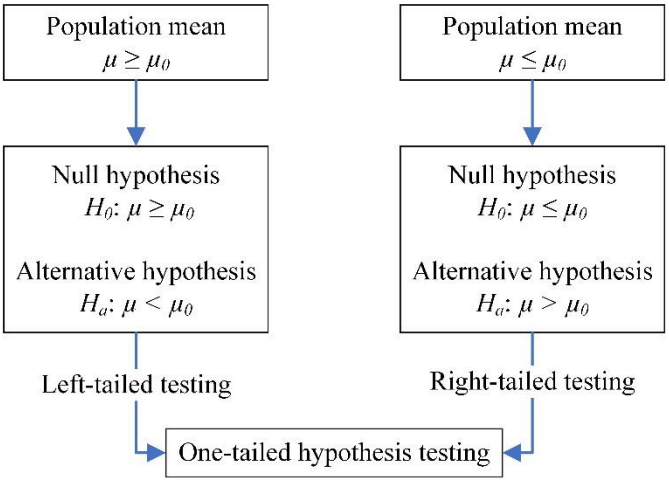

图 2. 均值的单尾假设检验

当零假设为真却被拒绝时，其概率称为显著性水平 α。显著性水平 α 决定概率分布曲线中临界区域或拒绝域的大小；在该区域内，零假设会被拒绝。临界区域的补集称为接受域。在双尾假设检验中，临界区域位于两侧。 如图 3 所示，概率分布曲线两侧各占据 $\alpha/2$ 的面积。在单尾假设检验中，临界区域的面积为 α，可位于概率分布曲线的左侧或右侧，如图 4 所示。此外，将临界区域与接受域分隔开的数值称为临界值。已知样本量 n 和显著性水平 α 后，即可确定临界值。

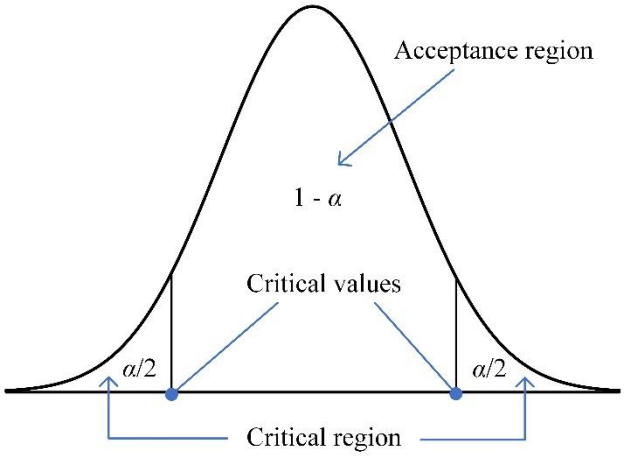

图 3. 双尾假设检验中的临界区域

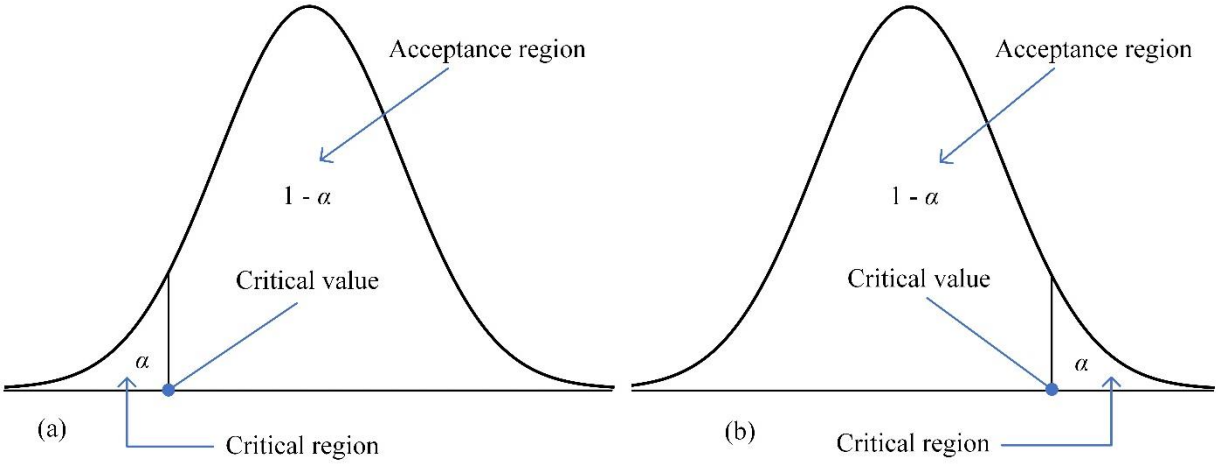

图 4. 临 界区域：(a) 左尾假设检验和 (b) 右尾假设检验中 g

2.2. 正态分布和 t分布

在统计学中，t分布 [44] 用于描述正态分布总体小样本的概率分布；当样本量较大时，它会趋近于正态分布。本研究将t分布用于假设检验。确定临界值需要样本量 n 和显著性水平 α 。图5展示了正态分布与t分布之间的差异。使用t分布时，双尾假设检验的临界值为 $-t_{\alpha/2}^{(n-1)}$ 和 $t_{\alpha/2}^{(n-1)}$ 。此外，对于左尾和右尾假设检验，临界值分别为 $-t_{\alpha}^{(n-1)}$ 和 $t_{\alpha}^{(n-1)}$ 。

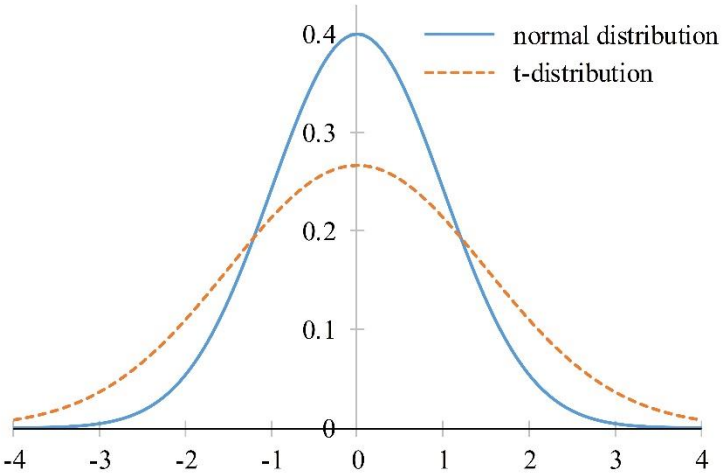

图 5. 正态分布和 t 分布

2.3. 统计表述

用于假设检验的t分布检验统计量记为 t，表示为

$$t = \frac{\bar{m} - \mu_0}{\frac{s}{\sqrt{n}}} \tag{1}$$

其中，$\bar{m}$ 是样本均值，s 是样本标准差。当 t落入临界区域时，拒绝零假设。对于双尾假设检验，当 t 的值落在 $(-\infty, -t_{\alpha/2}^{(n-1)})$ 或 $(+t_{\alpha/2}^{(n-1)}, +\infty)$ 中时，拒绝零假设。对于左尾和右尾假设检验，当 t 的值分别落在 $(-\infty, -t_{\alpha}^{(n-1)})$ 和 $(+t_{\alpha}^{(n-1)}, +\infty)$ 中时，拒绝零假设。

3. 萤火虫算法

萤火虫算法是自然启发算法 [16, 45]，其模拟了萤火虫的社会行为。这些算法基于以下概念:

任意两只萤火虫中，较差的一只会向较好的一只移动。如果没有更好的萤火虫，萤火虫就会随机移动。萤火虫的优劣根据问题的目标函数和约束来确定。

较差萤火虫向较优萤火虫移动的距离取决于较优萤火虫的吸引度。萤火虫的吸引度会随着与另一只萤火虫之间距离的增大而降低。

吸引　　iveness β 决定了一只萤火虫吸引其他萤火虫的强度，其定义为 [46　　]

$$\beta(r) = \beta_0 e^{-\gamma r^2} \tag{2}$$

其中，r 是到萤火虫的距离，β_0 是在$r = 0$处的吸引度。此外，γ 是固定的光吸收系数。

通常情况下，被另一只更明亮的萤火虫 j吸引的萤火虫 i，其位置通过以下更新方程进行更新，即

$$x_{i,k} = x_{i,k} + \beta(r_{ij})\left(x_{j,k} - x_{i,k}\right) + \varepsilon\left(rnd_{i,k} - \frac{1}{2}\right). \tag{3}$$

其中，$x_{i,\ k}$ 和$x_{j,\ k}$ 分别是萤火虫i 和j的k^{th} 分量。此外，r_{ij} 是两只萤火虫之间的距离。等式右侧的第三项是随机化项，其中ε 是一个系数，$rnd_{i,\ k}$ 是[0,1]上的一个随机数。

许多研究工作[47-51] 对传统萤火虫算法进行了进一步改进，以适应各种现实问题。本研究中的更新方程是结合[45] 和[47], 中的版本修改而成的混合版本，其形式为

$$x_{i,k} = x_{i,k} + \beta(\hat{r}_{ij})rnd_{i,k}\left(x_{j,k} - x_{i,k}\right) + \varphi_{i,k}\omega^{\tau}\left(rnd_{i,k} - \frac{1}{2}\right). \tag{4}$$

上述方程最初基于文献[45]中的研究。等式右侧的第二项和第三项分别是吸引度项和随机化项。本研究中，吸引度项乘以[0,1]上的一个随机数。此外，两只萤火虫之间的距离被归一化距离 $\hat{r}_{ij}$ [47]取代，其定义为

$$\hat{r}_{ij} = \frac{r_{ij}}{r_{max}}. \tag{5}$$

其中，r_{max} 是任意两只萤火虫之间可能的最大距离，表示为

$$r_{max} = \sqrt{\sum_{k=1}^{D}(up_k - low_k)^2} \tag{6}$$

其中，D 是设计变量的数量。此外，up_k 和low_k 分别是第k^{th} 个设计变量的上界和下界。归一化距离$\hat{r}_{ij}$ 是必要的，因为在高维大规模问题中，群体中的萤火虫可能彼此相距甚远。由于吸引度与每对萤火虫之间的距离成反比，因此使用未经归一化的距离时，算法的性能可能较差。此外，式（4）中的ω 是冷却因子[45]，而τ 表示当前迭代次数。最后，$\varphi_{i,k}$ 是初始随机性缩放因子[45],，其表示为

$$\varphi_{i,k} = \theta_{i,k}(up_k - low_k). \tag{7}$$

这里，$\theta_{i,\ k}$ 是一个用户定义系数，在本研究中，它是一个介于 [0,1] 之间的随机数。请注意，本研究的目标并非开发一种新的萤火虫算法。事实上，可以采用任何表现良好的萤火虫算法版本。

4. 统计萤火虫算法

在萤火虫算法中，萤火虫 i 会与萤火虫 j 进行比较。如果萤火虫 j 优于萤火虫 i，萤火虫 i 就会向萤火虫 j 移动。这个过程体现了两只萤火虫之间的协作。当萤火虫 i 向更优的萤火虫 j 移动时，并不能保证萤火虫 i 会因此变得更优。在本研究中，通过持续考察萤火虫对协作过程中的移动结果，收集其协作质量信息。每当萤火虫 i 向萤火虫 j 移动时，如果这次移动使萤火虫 i 到达更优的位置，就记为 1 分。反之，如果移动未使萤火虫 i 到达更优的位置，则记为 0 分。为启动假设检验，假设在算法开始之前，每只萤火虫 i 都已向萤火虫 j 完成一次成功的虚拟移动，因此初始记为 1 分，初始移动次数为 1。仅经过一次实际移动后，就可以计算协作得分的均值和标准差。如果均值较高，说明萤火虫 i 与萤火虫 j 的协作通常是有益的，因此萤火虫 i 与萤火虫 j 未来的协作也很可能有益。

在本研究中，若要认定萤火虫 i 与萤火虫 j 之间的协作可能有用，则协作得分的均值（记为 μ_{ij}）必须高于某个特定值 μ_o。这一表述可以通过假设检验来表达。因此，本次假设检验的目标是检查 μ_{ij} 是否大于或等于 μ_o，并采用特定的显著性水平 α。这是左尾假设检验，其中零假设 H_0 为 $\mu_{ij} \geq \mu_o$，备择假设 H_a 为 $\mu_{ij} < \mu_o$。如果检验统计量的值落在区间 $(-\infty, -t_\alpha^{(n-1)})$ 内，则拒绝零假设。若零假设被拒绝，则认为萤火虫 i 与萤火虫 j 之间的协作无效，其即将进行的

移动将被取消。若零假设未被拒绝，则允许萤火虫 i 向萤火虫 j移动。由于协作得分只能是 0 或 1，协作得分的均值位于 [0,1]上。本研究中，μ_o 的值设为 [0,1]上的一个随机数。

用于最小化问题的 SFA 机制伪代码如下。在伪代码中，$f(\mathbf{x})$表示待最小化的目标函数，而 $\mathbf{x}_i$ 表示萤火虫 i。此外，P_{ij} 表示萤火虫 i 与萤火虫 j协作的假设检验结果。当 P_{ij} =1 时，表示萤火虫 i 与萤火虫 j 的协作被认为是有效的，因此可以继续萤火虫 i 即将进行的移动。

用于最小化问题的 SFA:

生成由 $npop$ 只萤火虫组成的初始群体；确定 $f(\mathbf{x}_i)$、 $i = 1$、……、 $npop$；将每一对萤火虫有序对的初始移动次数设为 1；将每一对萤火虫有序对的协作得分设为 1；设置 $P_{ij} = 1$ 、 i、 $j = 1$、……、 $npop$；$\tau = 0$；当 $(\tau < max\ iteration)$ 时，执行：对于 $i = 1$ 到 $npop$，对于 $j = 1$ 到 $npop$，如果 $(f(\mathbf{x}_j) < f(\mathbf{x}_i))$ // 萤火虫 j 优于萤火虫 i如果 $(P_{ij} = 1)$ 将萤火虫 i 移向萤火虫 j ；确定 $f(\mathbf{x}_i)$ 并收集协作得分；结束如果 随机获得 μ_0 ，其取值范围为 [0,1]；执行假设检验以获得 P_{ij}；结束如果 结束循环 结束循环 对萤火虫进行排序并找出当前最优解；$\tau = \tau + 1$；结束循环

5. 桁架拓扑优化

本研究中，桁架拓扑优化的目标是通过调整其单元的截面积，使桁架的重量最小化。单元的面积可以为零，这表示该单元不存在。本研究考虑的桁架优化问题通常可表示为

最小化

$$W = \sum_{i=1}^{EL} \rho_i L_i A_i \tag{8}$$

约束条件

桁架在运动学上稳定

$$|\sigma_i| \le |\sigma_{a,i}|, \quad i = 1,2, \dots, el$$
$$|\delta_j| \le |\delta_{a,j}|, \quad j = 1,2, \dots, ndof \tag{9}$$

其中

W 桁架的重量EL 地面结构中所有可能单元的数量el 桁架中单元的数量$ndof$ 桁架的位移自由度数量ρ_i 单元的重量密度 i 个EL 单元L_i 单元的长度 i 个EL 单元A_i 单元的面积 i 个EL 单元σ_i 单元的应力 i 个el 桁架中的单元$\sigma_{a,\ i}$ 单元的许用应力 $i\delta_j$ 位移自由度 j 个$ndof$ 桁架的位移自由度$\delta_{a,\ j}$ 位移自由度的许用位移 j。

当桁架满足所有约束时，该桁架为可行桁架；否则，该桁架为不可行桁架。对于所提出的算法，优化问题式（8）可写为

最小化

$$f = \begin{cases} W, & \text{若桁架可行} \\ 10^{20}, & \text{若桁架不可行。} \end{cases} \tag{10}$$

在一些研究中，[38, 52, 53], 仅为运动学不稳定桁架的目标函数赋予较大的正值。由于萤火虫算法具有稳健的全群体信息交换机制，整个群体只需经过少数几次迭代，便可轻松移动到可行区域。因此，在本研究中，通过为任何不可行桁架的目标函数赋予较大的正值，简化了对搜索解的评估。

除了式（9）中的约束参数外，还必须为每根桁架杆件设定最小面积 A_{min} 和最大面积 A_{max}。如果面积设计变量的值低于 A_{min}，则将面积直接设为零。此外，如果面积设计变量的值高于 A_{max}，则将面积设为 A_{max}。这种解码方式在桁架拓扑优化中很常见。不过，本研究中的算法并不直接使用通过上述简单解码过程得到的拓扑结构，而是通过单元删除算法 [38] 对得到的拓扑结构进行改进，以删除其中的冗余单元，从而获得更优的拓扑结构。冗余单元是指明显不稳定或明显不受力的单元。单元删除算法的详细信息见 [38]。

6. 数值结果

通过求解四个桁架拓扑优化问题，对 SFA 进行了验证。这些问题也由萤火虫算法求解，其更新方程见式（4）。我们比较了两种算法得到的结果，以考察 SFA 如何通过运行时间和目标函数评估次数降低计算开销，以及与萤火虫算法相比，SFA 如何保持所得结果的质量。算法系数设置如下：$\beta_0 = 2, \gamma = 1, \omega$ =0.978 和 $\theta_{i,\ k}$ = 在 [0,1] 上取随机值。此外，对于每个问题，两种算法均运行 1000 次，以考察所得结果的质量和均匀性。选取显著性水平 $\alpha = 0.05$。每次运行包含 1000 次迭代。因此，某一对萤火虫之间的移动次数可能在 1 到 1000 之间变化。为简单起见，采用固定临界值 $-t_{\alpha}^{(500-1)} = -1.65$，这相当于固定样本量为 500。对于每个问题，萤火虫算法均以包含 20 只萤火虫的群体运行。而 SFA 则分别采用 20、30 和 40 只萤火虫的群体，以考察 SFA 的行为。使用以下缩写，即 AMPDE(10×10)、SCGA(50)、粒子群优化(2×10)、FA(20)、

SFA(20)、SFA(30) 和 SFA(40)，括号中的数字表示群体规模。两种算法的程序均使用 C++ 编写，并在配备 Intel Core i9-7900X 处理器和 128 GB RAM 的计算机上运行。

6.1. 问题*1*：*12*节点、*39*单元桁架

第一个问题是图6所示的二维问题，许多研究人员已对其进行研究 [38, 53-55]。所采用的地面结构有12个节点，且仅考虑图中所示的39个单元。本研究不假设地面结构沿中间垂直线对称，因此面积设计变量的数量为39。图6还展示了力、支座和问题尺寸。采用的杨氏模量为 $E = 10^4$ ksi，重量密度为 $\rho = 0.1$ lb/in^3 。此外，采用的许用应力为$\sigma_a = \pm 20$ ksi，许用位移为 $\delta_a = \pm 2$ in。单元面积 A 的取值范围设为0.05至2.25 in^2, ，而单元的面积设计变量取值范围设为 -2.25至2.25 in^2。若单元的面积设计变量小于0.05 in^2, ，则认为其面积为零。相反，若单元的面积设计变量大于2.25 in^2, ，则将其面积设为2.25 in^2。

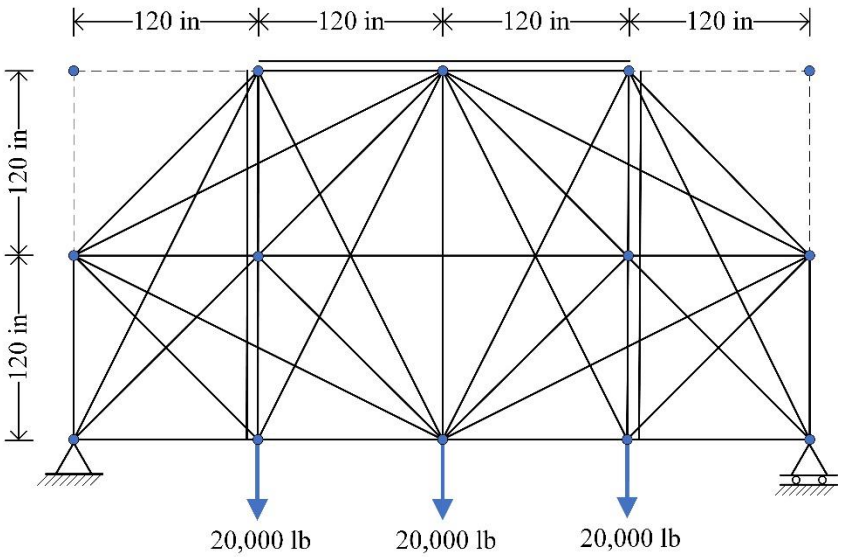

图6。 问题1：地面结构

两种算法在所有算例中得到的最佳拓扑相同，如图7所示。两种算法得到的单元面积和最小重量也相同，如表1所示。值得注意的是，已有多位研究人员得到相同的最佳拓扑 [38, 53-55]。此外，许多研究得到的最小重量均为193.200磅[38, 53]。[55], 中得到的最小重量与之相近，为193.012磅，而[54] 报告的数值为193.547磅。值得注意的是，这些研究工作[38, 53-

55] 假设地面结构沿中间垂直线对称，从而将面积设计变量的数量从39个减少到21个。

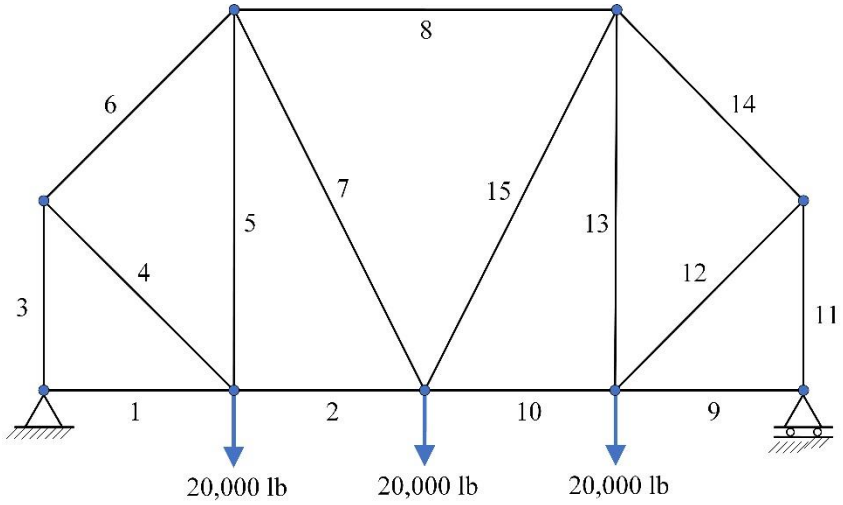

图7. 问题1：最佳拓扑

表1. 问题1：最优解的单元面积

单元	Shakya等人 [38][a]	Wu 和 Tseng [53][a]	本研究
	面积（in^2）		
1,9	0.050	0.050	0.050
2,10	0.750	0.750	0.750
3,11	1.500	1.500	1.500
4,12	1.061	1.060	1.061
5,13	0.250	0.250	0.250
6,14	1.061	1.061	1.061
7,15	0.559	0.559	0.559
8	1.000	1.000	1.000
重量（磅）	193.200	193.199	193.200
最大应力（ksi）	20.000	20.000	20.000
最大位移（in）	1.440	1.440	1.440

[a] 假设具有对称性

表2展示了所得结果的统计结果。表中清楚地表明，与FA相比，SFA的三种情形所需的计算开销小得多，这一点可从目标函数评估次数和运行时间看出。尽管SFA(20)求解该问题所需的计算开销小于FA(20)，但其整体性能不如FA(20)。然而，SFA(30)的整体性能与FA(20)相当，同时其计算开销几乎少了3.0倍。此外，SFA(40)在所有统计结果上均优于FA(20)，且所需的计算开销也更小。所有情形下的最佳收敛结果见图8。尽管SFA大幅减少了计算量，但仍取得了良好的收敛结果，与FA的收敛情况相当。

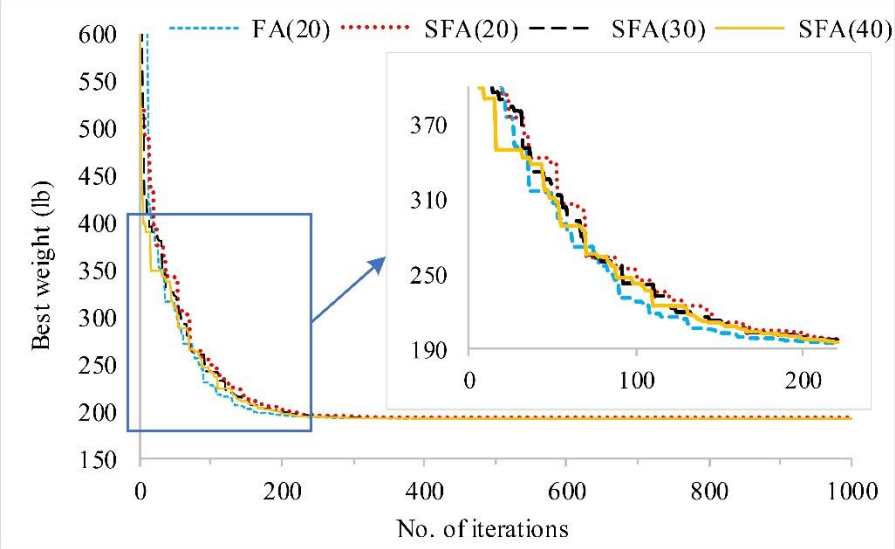

图8. 问题1：目标函数值的最佳收敛结果

表2. 问题1：1000次运行解的统计学

	Shakya等人 [38]	Wu 和 Tseng [53]	本研究			
算法	粒子群优化(2×10)	AMPDE(10×10)	FA(20)	SFA(20)	SFA(30)	SFA(40)
最小重量（磅）	193.200	193.199	193.200	193.200	193.200	193.200
平均重量（磅）	–	–	216.0	218.9	214.7	213.2
最大重量（磅）	–	–	304.6	318.8	352.3	275.6
标准差（磅）	–	–	15.6	18.9	15.5	14.6
运行解的百分比不比.....重的	–	–	21.5	19.2	24.2	28.4

最小　　整个……的重量 1000 次运行中，超过 2% 的（%）						
目标函数评估次数的平均值 目标函数评估次数	20000	–	189822	35242	70639	116917
平均有效 协作次数	–	–	3983	3707	5338	7035
平均运行时间（秒）	–	–	32.9	7.8	14.3	22.8

6.2. 问题*2*：*10*节点、*45*单元桁架

第二个问题是图9所示的二维问题。与第一个问题类似，许多研究人员也研究过这个问题[52, 56-58]。由2×5 网格构建的地面结构有10个节点。本研究未采用地面结构沿中间垂直线的对称性。由于考虑了连接地面结构全部10个节点的所有45个单元，因此面积设计变量的数量为45。图9还展示了力、支座和问题尺寸。采用的杨氏模量为E =10^4 ksi，重量密度为 ρ = 0.1 lb/in^3 。此外，采用的许用应力为 σ_a =± 25 ksi，许用位移为δ_a = ± 2 in。单元的面积A 在0.09至1.00 in^{2}之间变化。将单元的面积设计变量设定为在−1.00至1.00 in^{2}之间变化。如果单元的面积设计变量小于0.09 in^2, ，则将其面积视为零。如果单元的面积设计变量大于1.00 in^2, ，则将面积设为1.00 in^2。

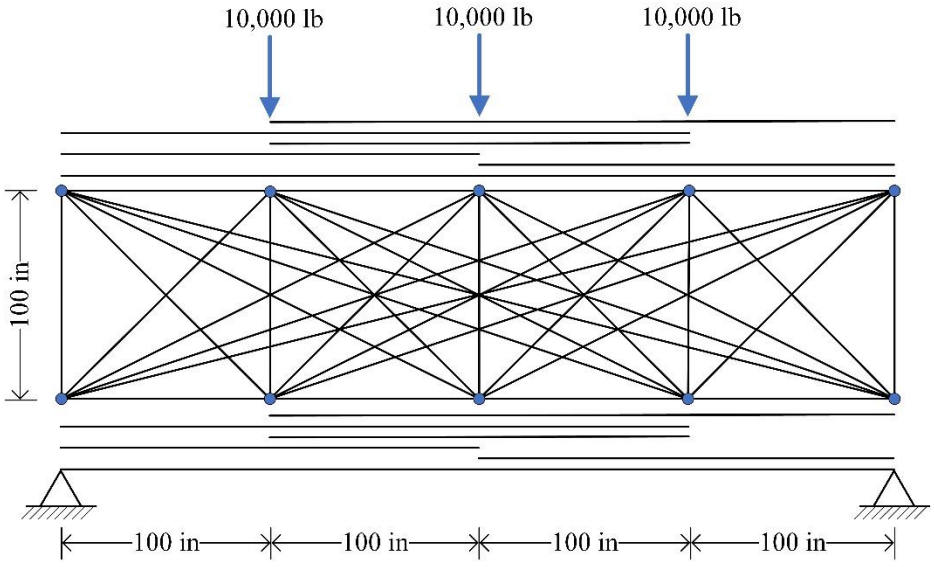

图9. 问题2：地面结构

两种算法在所有算例中得到的最佳拓扑结构相同，如图10所示。两种算法得到的单元面积和最小重量也相同，如表3所示。值得注意的是，多位研究人员也得到了相同的最佳拓扑结构 [56-58]。此外，[56, 57] 中得到了相同的最小重量 44.000 lb，而 [52] 中得到的最小重量 44.033 lb 与之相近。

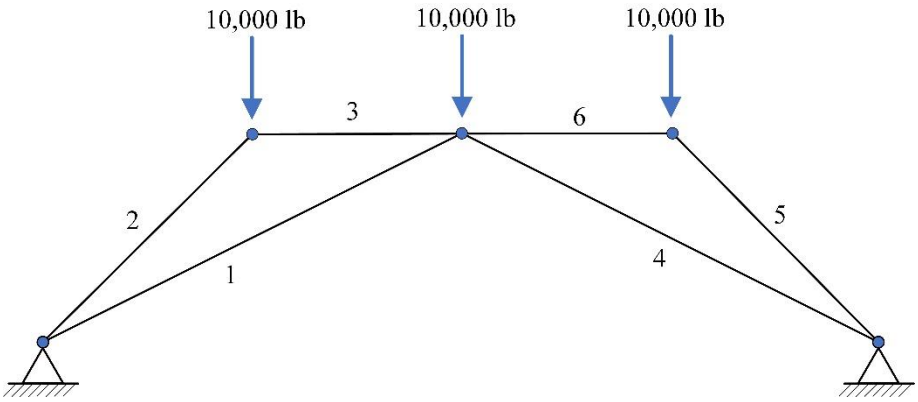

图10. 问题2：最佳拓扑

表3. 问题2：最优解的单元面积

	Faramarzi 和 Afshar [56]	Li [57]	本研究
单元	面积 (in^2)		
1	0.447	0.447	0.447
2	0.566	0.566	0.566
3	0.400	0.400	0.400
4	0.447	0.447	0.447
5	0.566	0.566	0.566
6	0.400	0.400	0.400
重量 (磅)	44.000	44.000	44.000
最大应力 (ksi)	25.000	25.000	25.000
最大位移 (in)	1.250	1.250	1.250

表4展示了所得结果的统计学数据。同样，SFA 的三种情况所需的计算开销都远小于 FA。使用 SFA(20) 求解问题的计算开销约为 FA(20) 的六分之一。然而，其整体性能仅略逊于 FA(20)。此外，SFA(30) 的表现略优于 FA(20)，而计算开销约为其三分之一。此外，SFA(40) 的计算开销也小于

FA(20)，且在所有统计结果上都优于 FA(20)。所有情况的最佳收敛结果如图11所示。尽管计算开销较小，SFA 仍能取得良好的收敛情况，与 FA 的收敛结果相当。

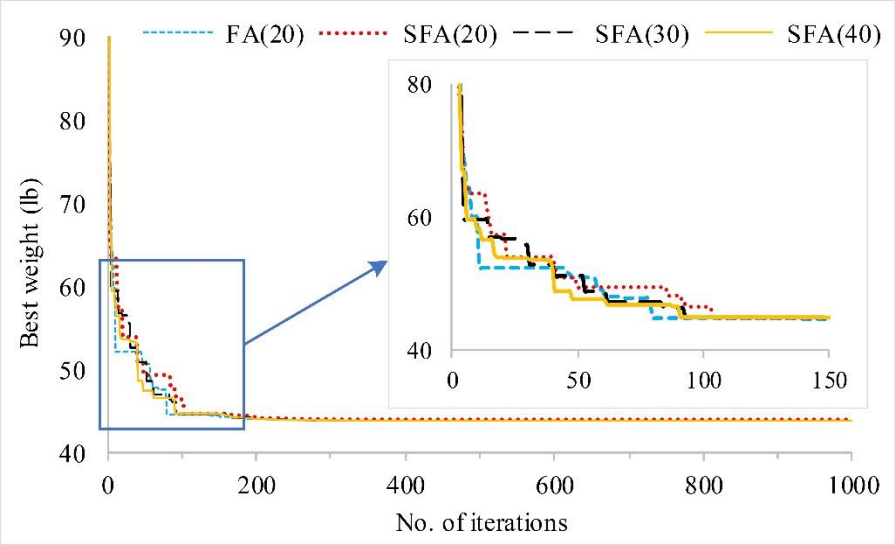

图11. 问题2：目标函数值的最佳收敛结果

表4. 问题2：1000次运行解的统计学

	Faramarzi 和 Afshar [56]	Li [57]	本研究			
算法	CA-LP	SCGA(50)	萤火虫算法(20)	SFA(20)	SFA(30)	SFA(40)
最小重量（磅）	44.000	44.000	44.000	44.000	44.000	44.000
平均重量（磅）	–	–	44.4	44.6	44.3	44.1
最大重量（磅）	–	–	64.9	91.6	60.9	52.0
标准差（磅）	–	–	1.1	2.4	0.8	0.4
运行解中不重于以下最小重量的解所占百分比 1000 次运行的最小重量，且大于 2% 的百分比（%）	–	–	68.8	63.1	72.1	86.5
目标函数平均计算次数 函数评估次数	–	–	190095	31284	64157	107567

有效协作次数的平均值	–	–	2492	2230	3396	4581
协作 平均运行时间（秒）	–	–	42.5	7.0	14.5	24.4

6.3. 问题*3*：*35*节点、*595*单元桁架

第三个问题是图12 [56]中的一个二维问题。本研究将地面结构中的节点数从[56], 中使用的12个增加到图12中的35个。请注意，网格间距均匀。考虑地面结构35个节点之间完全连接的全部595个单元。假设地面结构沿中间水平线对称。因此，面积设计变量的数量从595个减少到315个。图12还展示了力、支座和问题尺寸。采用杨氏模量 $E = 10^4$ ksi和重量密度 $\rho = 0.1$磅/英寸^3 。此外，采用许用应力 $\sigma_a = \pm 5$ ksi和许用位移 $\delta_a = \pm 0.6$英寸。单元面积 A 设为在0.09至35.00英寸^2, 之间变化，而单元的面积设计变量设为在 -35.00至35.00英寸^{2}之间变化。如果单元的面积设计变量小于0.09英寸^2, ，则将面积视为零。如果单元的面积设计变量大于35.00英寸^2, , 则将面积设为35.00英寸^2。

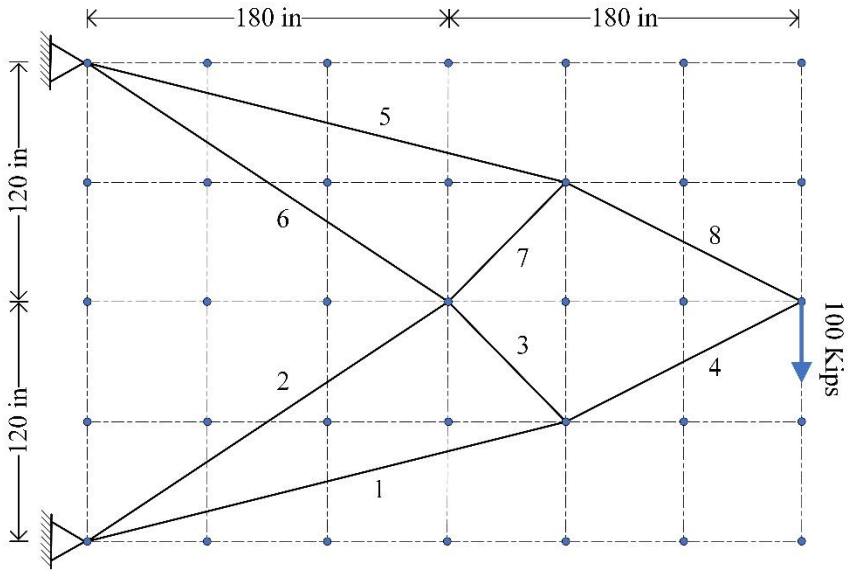

图12. 问题3：地面结构和最佳拓扑

本研究中两种算法所有案例得到的最佳拓扑结构相同，如图12所示。两种算法得到的单元面积和最小重量也相同，如表5所示。最小重量为2232.000磅，优于[56], 中得到的2400.001磅。

表5. 问题3：最优解的单元面积

单元	面积（in^2）[a]
1, 5	24.739
2, 6	7.211
3, 7	5.657
4, 8	22.361
重量（磅）	2232.000
最大应力（ksi）	5.000
最大位移（in）	0.558

[a]假定为对称性

表6展示了所得结果的统计结果。可以发现，SFA 的三种情况所需的计算开销远小于 FA。尽管使用 FA(20) 求解该问题的计算开销远大于 SFA(20) 和 SFA(30)，但其整体性能优于 SFA(20) 和 SFA(30) 的整体性能。此外，SFA(40) 的计算开销约比 FA(20) 小 1.5 倍，同时在所有统计结果中，其性能仍略优于 FA(20)。所有情况下的最佳收敛结果如图13所示。尽管 SFA 所需的计算开销远小于 FA，但 SFA 仍能获得与 FA 相当的良好收敛情况。

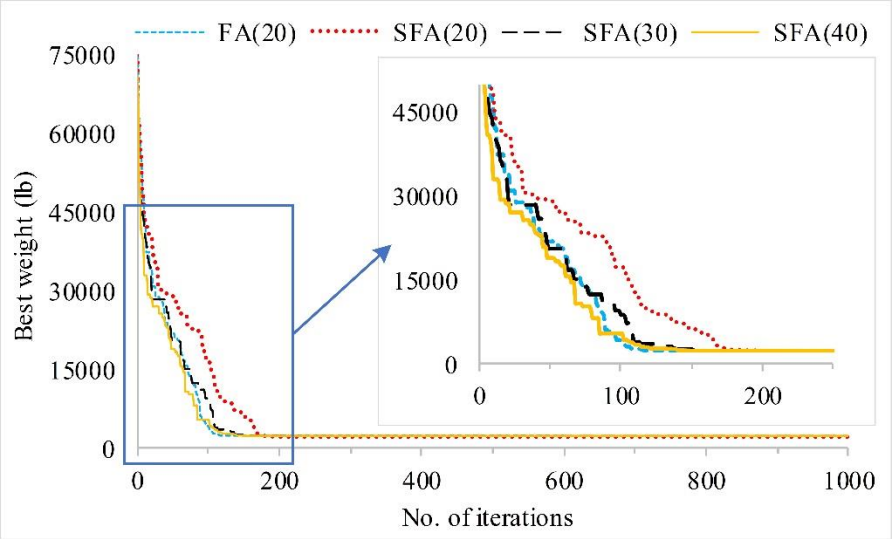

图13. 问题3：目标函数值的最佳收敛结果

表6. 问题3：1000次运行解的统计学

	Faramarzi 和 Afshar [56]	本研究			
算法	元胞自动机	FA(20)	SFA(20)	SFA(30)	SFA(40)
最小重量（磅）	2400.0	2232.0	2232.0	2232.0	2232.0
平均重量（磅）	–	2766.8	3281.4	2920.1	2757.4
最大重量（磅）	–	4640.2	5785.7	5039.7	4568.1
标准差（磅）	–	420.3	642.5	489.1	412.9
百分比 of the run 重量不超过 最低重量的解 全部 1000 次运行中，由 大于 2% (%)	–	2.3	0.6	1.3	2.6
目标函数评估次数平均值 函数评估次数	–	189338	41492	74496	117923
有效协作次数平均值 协作	–	6341	5112	5057	5608
平均运行时间（秒）	–	467.2	116.6	200.5	317.8

6.4. 问题*4*：*18*节点、*153*单元桁架

最后一个问题是图14所示的三维问题，相关研究见 [38]。地面结构包含18个节点，由3×3×2 网格构成。考虑18个节点之间完全连接的全部153个单元。由于不假设对称性，面积设计变量的数量为153。图14还展示了力、支座和问题尺寸。采用杨氏模量 $E = 200$ GPa 和重量密度 ρ =77 kN/m^3 。此外，采用许用应力 $\sigma_a = \pm 150$ MPa 和许用位移 $\delta_a = \pm 20$ mm。单元的面积 A 设定为在0至2000 mm^2, 之间变化，而单元的面积设计变量设定为在 -2000 至2000 mm^{2}之间变化。如果单元的面积设计变量小于或等于0 mm^2, ，则该面积视为零。相反，如果单元的面积设计变量大于2000 mm^2, ，则该面积设为2000 mm^2。

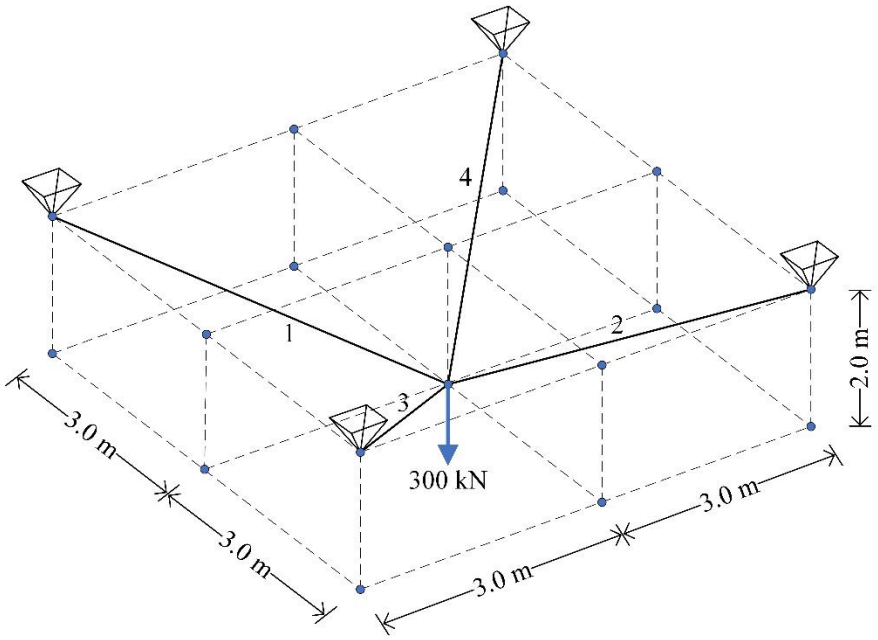

图14. 问题4：地面结构与最佳拓扑

两种算法在所有算例中得到的最佳拓扑结构相同，如图14所示。两种算法得到的最小重量也相同，均为 1694.0 N。在最优解中，四根杆件的面积并非总是相同。然而，单元1和2的面积始终相同，单元3和4的面积也始终相同。值得注意的是，在 [38] 中也得到了相同的最佳拓扑和 1694.0 N 的最小重量。

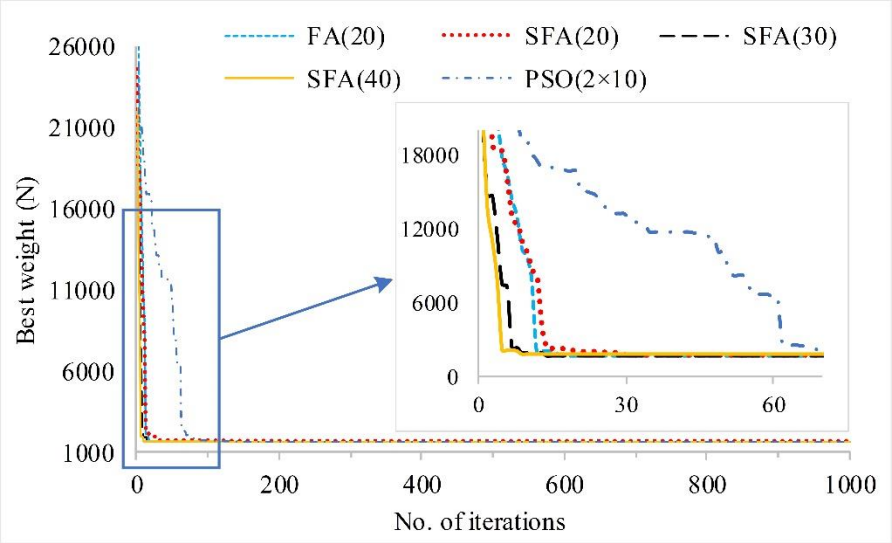

图15. 问题4：目标函数值的最佳收敛结果

表7. 问题4：1000次运行解的统计学

	Shakya等人 [38]	本研究			
算法	粒子群优化(2×10)	FA(20)	SFA(20)	SFA(30)	SFA(40)
最小重量 (N)	1694.0	1694.0	1694.0	1694.0	1694.0
平均重量 (N)	2194.8	1751.1	1774.1	1707.3	1695.8
最大重量 (N)	8976.5	3738.9	5756.8	2910.5	2442.3
标准差 (N)	986.2	239.4	313.3	114.2	33.6
运行解的百分比	69.6	93.8	92.5	98.6	99.7
不超过最小					
整个 1000 次运行的重量					
大于 2% (%)					
平均目标函数	20000	189910	31018	62663	104774
评估次数					
平均有效	-	1950	1930	2763	3643
协作次数					
平均运行时间（秒）	-	179.4	30.1	61.1	102.1

表7展示了所得结果的统计数据。可以看出，萤火虫算法和 SFA 得到的统计结果远优于 [38], 中采用多种群粒子群优化[59-61] 所得的结果。尽管 FA(20) 的表现优于 PSO(2×10)，但它也需要更多

计算开销更大。在目标函数评估次数的平均值较为合理的情况下，SFA(20) 的表现远优于 PSO(2×10)。表7清楚地表明，SFA 可以大幅降低传统 FA 的计算开销。使用 SFA(20) 求解该问题的计算开销几乎比使用 FA(20) 小 6 倍。不过，其整体性能不如 FA(20)。尽管如此，两者的差异并不显著。此外，在所有统计结果中，SFA(30) 和 SFA(40) 的表现都远优于 FA(20)，同时计算开销也更小。所有情况的最佳收敛结果如图15所示。尽管 SFA 的计算开销比 FA 更小，但仍能获得良好的收敛结果，与 FA 得到的收敛结果相当。此外，FA 和 SFA 的收敛速度都远快于 [38] 中多种群粒子群优化所得到的收敛速度。

7. 结论

本研究提出了一种统计萤火虫算法（SFA）。在该算法中，优化过程中将均值假设检验应用于普通萤火虫算法的机制，以便将萤火虫的移动限制在仅可能有用的移动上。为了预测萤火虫对之间协作的潜在效用，算法会根据移动结果持续收集协作质量数据，随后计算这些结果的统计量。要判定任意一对萤火虫之间的协作是否有用，其协作的平均得分必须高于某个特定值。针对假设检验，将这一判定表述为检验命题，并利用过去协作结果的统计量。只有当每一对萤火虫均通过每项假设检验时，才能继续进行它们当前的协作。

本文提出的 SFA 用于求解四个桁架拓扑优化问题，以验证其有效性和效率。加入统计策略后，由于识别出潜在的无效协作，大量萤火虫的移动得以减少。随后，SFA 仅需比萤火虫算法少得多的计算开销，即可获得精度相近的结果。此外，本研究还表明，萤火虫算法和 SFA 的性能均优于粒子群优化。虽然萤火虫算法需要更大的计算开销，但其性能优于粒子群优化；而 SFA 则能以合理的计算开销取得更好的性能。SFA 还能使目标函数值快速收敛。因此，对于需要大量计算开销的大型复杂问题，SFA 可视为一种稳健的优化工具。

致谢

谨向东盟大学网络/东南亚工程教育发展网络（AUN/SEED-Net）和诗琳通国际理工学院为第一作者提供的奖学金致以诚挚谢意。

参考文献

[1] X.-S. Yang，《自然启发算法与应用优化》，施普林格，Cham，2018年。[2] V.C. SS，A. HS，面向优化问题的自然启发式元启发算法，《计算》，104 (2) (2022年) 251-269。[3] L. Mei，Q. Wang，土木工程中的结构优化：文献综述，《建筑》，11 (2) (2021) 66。[4] N.D. Lagaros，M. Papadrakakis，G. Kokossalakis，使用进化算法进行结构优化，《计算机与结构》，80 (7-8) (2002) 571-589。[5] M. Pant，H. Zaheer，L. Garcia-Hernandez，A. Abraham，差分进化：二十余年研究综述，《人工智能工程应用》，90 (2020年) 103479。[6] S. Das，S.S. Mullick，P.N. Suganthan，差分进化的最新进展——更新版综述，《群体与进化计算》，27 (2016年) 1-30。[7] K.V. Price，差分进化，载于：《优化手册：从经典方法到现代方法》，施普林格，柏林，2013年，187-214。[8] M.F. Ahmad，N.A.M. Isa，W.H. Lim，K.M. Ang，差分进化：基于前沿研究成果的最新综述，《亚历山大工程学杂志》，61 (5) (2022年) 3831-3872。[9] K.-F. Man，K.-S. Tang，S. Kwong，遗传算法：概念与应用 [于工程设计]，《IEEE工业电子学汇刊》，43 (5) (2002) 519-534。[10] S. Katoch，S.S. Chauhan，V. Kumar，遗传算法综述：过去、现在与未来，《多媒体工具与应用》，80 (5) (2021) 8091-8126。[11] B. Alhijawi，A. Awajan，遗传算法：理论、遗传算子、解与应用，《进化智能》，17 (3) (2024年) 1245-1256。[12] J. Kennedy，R. Eberhart，粒子群优化，ICNN'95神经网络国际会议，IEEE，珀斯，西澳大利亚州，澳大利亚，1995年，1942-1948。[13] A.G. Gad，粒子群优化算法及其应用：系统综述，《工程计算方法档案》，29 (5) (2022年) 2531-2561。

[14] T.M. Shami，A.A. El-Saleh，M. Alswaitti，Q. Al-Tashi，M.A. Summakieh，S. Mirjalili，粒子群优化：全面综述，IEEE Access，10 (2022年) 10031-10061。[15] W.-B. Du，Y. Gao，C. Liu，Z. Zheng，Z. Wang，适度即为更好：有限信息粒子群优化，《应用数学与计算》，268 (2015年) 832-838。[16] X.-S. Yang，A. Slowik，萤火虫算法，载于：《群体智能算法》，CRC出版社，2020年，163-174。[17] X.-S. Yang，连续优化的多目标萤火虫算法，《计算机辅助工程》，29 (2) (2013年) 175-184。[18] S.L. Tilahun，J.M.T. Ngnotchouye，连续型萤火虫算法：综述，《人工智能评论》，51 (3) (2019年) 445-492。[19] S.L. Tilahun，J.M.T. Ngnotchouye，离散优化问题的萤火虫算法：综述，《KSCE土木工程杂志》，21 (2) (2017年) 535-545。[20] R.V. Rao，教与学优化算法，载于：《教与学优化算法及其工程应用》，施普林格，Cham，2016年，9-39。[21] M. Dastan，S. Shojaee，S. Hamzehei-Javaran，V. Goodarzimehr，用于求解工程与数学问题的混合教与学优化，《巴西机械科学与工程学会杂志》，44 (9) (2022年) 431。[22] R. Xue，Z. Wu，教与学优化算法的应用与分类综述，IEEE Access，8 (2019) 1062-1079。[23] S. Kumar，G.G. Tejani，N. Pholdee，S. Bureerat，P. Jangir，用于结构优化的多目标教与学优化，《智慧科学》，10 (1) (2022年) 56-67。[24] M. Lohrer，萤火虫算法与粒子群优化的比较，奥克兰大学，美国，2013年。[25] A. Manguri，H. Hassan，N. Saeed，R. Jankowski，土木工程结构的拓扑、尺寸与形状优化：综述，《工程与科学计算机建模》，142 (2) (2025年) 933-971。[26] J. Zhou，Y. Zeng，G. Li，基于有限粒子法的桁架结构尺寸、形状与拓扑优化，《计算机与结构》，305 (2024年) 107570。[27] N. Petrović，N. Kostić，N. Marjanović，R.R. Nikolić，R. Ulewicz，评估基数约束对桁架尺寸、形状与拓扑同步优化的影响，《材料》，18 (7) (2025年) 1457。[28] A. Mortazavi，一种用于桁架结构拓扑与尺寸同步优化的新型二项式策略，《工程优化》，57 (5) (2025年) 1163-1197。

[29] K. Wansasueb, T. Kunakote, N. Panagant, N. Pholdee，桁架尺寸优化——基于梯度的优化器与元启发式算法的比较，Structural and Multidisciplinary Optimization，68 (7) (2025年) 1-18。[30] A. Kaveh, M.R. Dehkordi, A.H. Shaghaghian，基于神经梯度进化法的桁架结构尺寸与布局联合优化：变量联合域方法，Structures，爱思唯尔，2025年，109469。[31] R. Bodalal，采用基于堆的优化算法（HBO）对桁架结构进行尺寸与形状联合优化，Arabian Journal for Science and Engineering，50 (16) (2025年) 12689-12716。[32] S. Liu, H. Huang, S. Chen, Z. Dai, W. Li，桁架结构阻尼器布置、形状与尺寸的集成优化方法，《工程优化》，57 (9) (2025年) 2446-2466。[33] I.B. Ugur, S.O. Degertekin，基于循环系统的桁架结构形状与尺寸设计优化新方法，Applied Sciences，15 (7) (2025年) 3671。[34] D. Yu, Y. Wu, Z. Zhao, Q. Zhu，考虑长度约束的桁架结构拓扑优化方法，Structures，爱思唯尔，2025年，109079。[35] Q. Cai, J. Ma, Y.M. Xie, Y. Zhang, Y. Zhou，考虑整体稳定性和构件屈曲的桁架拓扑优化，Journal of Constructional Steel Research，226 (2025年) 109205。[36] Y. Lai, Q. Cai, Y. Li, J. Chen, Y.M. Xie，面向实际设计应用的桁架结构新型进化拓扑优化方法，Engineering Structures，324 (2025年) 119326。[37] G. Gao, Z.-y. Liu, Y.-b. Li, Y.-f. Qiao，生成桁架拓扑优化基结构的新方法，《工程优化》，49 (2) (2017年) 235-251。[38] A. Shakya, P. Nanakorn, W. Petprakob，采用单元移除算法的桁架拓扑优化基于基结构的表示，Structural and Multidisciplinary Optimization，58 (2) (2018年) 657-675。[39] P. Sahoo，概率与数理统计，，美国肯塔基州路易斯维尔40292，2013年。[40] A.F.M. Alkarkhi, W.A.A. Alqaraghuli，第5章——假设检验，载于：A.F.M. Alkarkhi, W.A.A. Alqaraghuli（编）Easy statistics for food science with R，学术出版社，2019年，53-80。[41] G. Casella, R. Berger，统计学中的假设检验，载于：N.J. Smelser, P.B. Baltes（编）International encyclopedia of the social & behavioral sciences，培格蒙，牛津，2001年，7118-7121。[42] U. Michelucci，假设检验，载于：Statistics for scientists: A concise guide for data-driven research，施普林格，2025年，129-135。[43] R.B. Davis, K.J. Mukamal，假设检验：均值，Circulation，114 (10) (2006年) 1078-1082。[44] R.B. Arellano-Valle, H. Bolfarine，t分布的若干特征，Statistics & Probability Letters，25 (1) (1995年) 79-85。

[45] X.-S. Yang, X. He，萤火虫算法：最新进展与应用，《国际群体智能杂志》，1 (1) (2013年) 36-50。[46] X.-S. Yang，自然启发式元启发式算法，Luniver press，英国，2010年。[47] S. Carbas，采用增强型萤火虫算法优化钢框架设计，《工程优化》，48 (12) (2016年) 2007-2025。[48] M.A. El-Shorbagy, A.M. El-Refaey，面向工程设计问题的混合遗传-萤火虫算法，《计算设计与工程杂志》，9 (2) (2022年) 706-730。[49] Z. Cheng, H. Song, D. Zheng, M. Zhou, K. Sun，一种采用性别区分新机制的混合萤火虫算法，用于全局优化，《应用专家系统》，224 (2023年) 120027。[50] W. Zhang, C. Jiao, Q. Zhou，基于性别差异、具有多重学习能力的萤火虫算法，《科学报告》，15 (1) (2025年) 28400。[51] A.K. Singh, A. Kumar，多目标：结合萤火虫算法的混合粒子群优化算法，采用带泄漏整流线性单元进行特征选择，《发现人工智能》，5 (1) (2025年) 192。[52] K. Deb, S. Gulati，采用遗传算法进行桁架结构最小重量设计，《分析与设计中的有限元》，37 (5) (2001年) 447-465。[53] C.-Y. Wu, K.-Y. Tseng，采用自适应多种群差分进化进行桁架结构优化，Structural and Multidisciplinary Optimization，42 (4) (2010年) 575-590。[54] L.F.F. Miguel, R.H. Lopez, L.F.F. Miguel，采用萤火虫算法对桁架结构进行多模态尺寸、形状和拓扑优化，《工程软件进展》，56 (2013年) 23-37。[55] G.-C. Luh, C.-Y. Lin，采用蚁群算法优化桁架结构设计，Structural and Multidisciplinary Optimization，36 (4) (2008年) 365-379。[56] A. Faramarzi, M. Afshar，将元胞自动机应用于桁架结构的尺寸与拓扑优化，《伊朗科学》，19 (3) (2012年) 373-380。[57] J.-P. Li，采用改进的物种保护遗传算法进行桁架拓扑优化，《工程优化》，47 (1) (2015年) 107-128。[58] J. Mockus, R. Paulavičius, D. Rusakevičius, D. Šešok, J. Žilinskas，将简化集帕累托-利普希茨优化应用于桁架优化，《全球优化杂志》，67 (1-2) (2017年) 425-450。[59] F. Kılıç, Y. Kaya, S. Yildirim，一种用于特征选择的新型多种群粒子群优化算法，《知识型系统》，219 (2021年) 106894。[60] G. Li, W. Wang, W. Zhang, Z. Wang, H. Tu, W. You，基于网格搜索的多种群粒子群优化算法，用于多模态多目标优化，《群体与进化计算》，62 (2021年) 100843。

[61] P. Zhang、J. Gao、L. Tan、K. Liu、K.I. Kostromitin、N. Kumar，云环境中用于工作流调度的遗传改进的多群体粒子群优化，《群体与进化计算》，98（2025年）102113。