Новый алгоритм метагеуристической оптимизации, называемый Cuckoo Search (CS), недавно был разработан Yang and Deb (2009). В этом документе представлено более подробное сравнительное исследование с использованием некоторых стандартных тестовых функций и недавно разработанных стохастических тестовых функций. Затем мы применяем алгоритм CS для решения задач оптимизации проектирования, включая конструкцию пружин и конструкций сварных балок. Оптимальные решения, полученные CS, намного лучше, чем лучшие решения, получаемые эффективным оптимизатором рой частиц. Мы обсудим уникальные функции поиска, используемые в CS, и последствия для дальнейших исследований.

Для простоты описания нового Cuckoo Search (Yang and Deb 2009) мы теперь используем следующие три идеализированных правила:

• Каждая кукушка закладывает по одному яйцу за раз, и сбрасывает его в случайно выбранном гнезде;

• Лучшие гнезда с высоким качеством яиц (растворы) переносятся на следующие поколения;

• Количество доступных гнезд хоста фиксировано, и хост может обнаружить чужеродное яйцо с вероятностью pa ∈ [0, 1]. В этом случае птица-хозяин может либо выбросить яйцо, либо покинуть гнездо, чтобы построить совершенно новое гнездо в новом месте.

Для простоты это последнее предположение может быть аппроксимировано фракцией pa тогдашних гнезд, заменяемых новыми гнездами (с новыми случайными решениями в новых местах). Для проблемы максимизации качество или пригодность решения может быть просто пропорционально целевой функции. Другие формы фитнеса могут быть определены аналогично функции фитнеса в генетических алгоритмах. Основываясь на этих трех правилах, основные этапы поиска кукушки (CS) можно суммировать как псевдокод, показанный на рисунке 1.

Здесь последовательные прыжки / шаги кукушки по существу образуют процесс случайного блуждания, который подчиняется степенному распределению длины с тяжелым хвостом.

Стоит отметить, что в реальном мире, если яйцо кукушки очень похоже на яйца хозяина, то яйцо кукушки с меньшей вероятностью будет обнаружено, поэтому фитнес должен быть связан с разницей в решениях. Поэтому неплохо провести произвольную прогулку с предубежденным способом с некоторыми случайными размерами шага. Демо-версия прилагается в Приложении (эта демонстрация не опубликована в фактической статье, а в качестве дополнения, чтобы помочь читателям правильно выполнить поиск кукушки).

Сравнительно легко реализовать алгоритм, и тогда нам нужно сравнить его с помощью тестовых функций с аналитическими или известными решениями. Существует множество контрольных тестовых функций, и нет стандартного списка или коллекции, хотя в литературе существуют обширные описания различных функций (Floudas et al 1999, Hedar 2005, Molga и Smutnicki, 2005). Например, тестовая функция Михалевича имеет много локальных оптимумов

Глобальный оптимум в 2D можно легко найти, используя Cuckoo Search, и результаты показаны на рисунке 3, где конечные местоположения гнезд отмечены значком ⋄. Здесь мы использовали n = 20 гнезд, α = 1 и pa = 0,25. Из рисунка видно, что, по мере приближения оптимального подхода, большинство гнезд сводятся к глобальному оптимуму. В различных симуляциях мы также замечаем, что гнезда также распределяются на разных (локальных) оптимумах в случае мультимодальных функций. Это означает, что CS может найти все оптимумы одновременно, если количество гнезд намного выше, чем количество локальных оптимумов. Это преимущество может стать более значительным при решении задач мультимодальной и многоцелевой оптимизации.

Мы также пытались варьировать количество гнезд хозяина (или размер популяции n) и вероятность pa. Мы использовали n = 5, 10, 15, 20, 50, 100, 150, 250, 500 и pa = 0, 0,01, 0,05, 0,1, 0,15, 0,2, 0,25, 0,4, 0,5. Из наших симуляций мы обнаружили, что n = 15-25 и pa = 0,15-0,30 достаточны для большинства задач оптимизации.

Различные тестовые функции в литературе предназначены для проверки эффективности алгоритмов оптимизации (Chattopadhyay 1971, Schoen 1993, Shang and Qiu 2006). Любой новый алгоритм оптимизации также должен быть проверен и протестирован против этих эталонных функций. В наших симуляциях мы использовали следующие тестовые функции.

Первая функция Де Йонга по существу является сферической функцией.

Общая схема оценки статистических характеристик эволюционных алгоритмов подробно обсуждалась Shilane et al (2008). Теперь мы можем сравнить поиск кукушки с PSO и генетические алгоритмы для различных тестовых функций. После реализации этих алгоритмов с использованием Matlab мы провели обширные симуляции, и каждый алгоритм выполнялся не менее 100 раз, чтобы провести значимый статистический анализ. Алгоритмы останавливаются, когда изменения значений функций меньше заданного допуска ǫ ≤ 10-5. Результаты суммированы в таблице 1, где цифры представлены в формате: среднее число оценок ± одно стандартное отклонение (коэффициент успеха), поэтому 3321 ± 519 (100%) означает, что среднее число (среднее) оценок функций равно 3321 с стандартное отклонение 519. Успешность поиска глобальных оптимумов для этого алгоритма составляет 100%. Функции, используемые в таблице: (1) Michaelwicz (d = 16), (2) Rosenrbrock (d = 16), (3) De Jong (d = 32), (4) Schwefel (d = 32), (5 (9) Первая стохастическая функция Янга, (10) Вторая стохастическая функция Янга, (11) Обобщенная функция Робенбрука со стохастическими компонентами и (11), (6) Растигин, (7) (12) Стохастическая функция Де Йонга

Из сравнительного исследования производительности CS с GA и PSO мы знаем, что наш новый поиск кукушки в сочетании с полетами L'evy очень эффективен и оказывается превосходным почти для всех тестовых задач. Отчасти это связано с тем, что в CS меньше параметров, чем в PSO и генетических алгоритмах.

На самом деле, помимо размера популяции n, существует по существу один параметр pa. Если мы внимательно рассмотрим алгоритм CS, есть по существу три компонента: выбор лучших, эксплуатация локальным случайным блужданием и исследование рандомизацией через рейсы L'evy по всему миру. Выбор лучшего, сохраняя лучшие гнезда или решения, эквивалентен некоторой форме элитарности, обычно используемой в генетических алгоритмах, которая гарантирует, что наилучшее решение будет передано на следующую итерацию, и нет риска, что лучшие решения будут выбрасываться из Население. Эксплуатация вокруг лучших решений выполняется с использованием локальной случайной ходьбы

Если εt подчиняется гауссовскому распределению, это действительно становится стандартным случайным блужданием. Это эквивалентно критическому этапу настройки высоты тона в поиске гармонии (Geem et al 2001, Yang 2009). Если εt берется из распределения L'evy, шаг перемещения больше и может быть потенциально более эффективным. Однако, если шаг слишком велик, существует риск, что перемещение слишком далеко. К счастью, элитарность, поддерживая лучшие решения, гарантирует, что действия по эксплуатации находятся в непосредственной близости от лучших решений на местном уровне. С другой стороны, чтобы эффективно пробовать пространство поиска, чтобы новые генерируемые решения были достаточно разнообразными, шаг исследования выполняется в терминах рейсов Л'ев. Напротив, большинство метаэвристических алгоритмов используют либо равномерные распределения, либо гауссовы для создания новых исследовательских движений (Geem et al 2001, Blum and Rilo 2003). Если пространство поиска велико, рейсы Л'еви обычно более эффективны. Таким образом, хорошая комбинация из трех вышеупомянутых компонентов может привести к эффективному алгоритму, таким как поиск кукушки.

Кроме того, наши симуляции также показывают, что скорость сходимости нечувствительна к параметрам, зависящим от алгоритма, таким как pa. Это также означает, что нам не нужно точно настраивать эти параметры для конкретной проблемы. Впоследствии CS является более общим и надежным для многих проблем оптимизации, по сравнению с другими метаэвристическими алгоритмами.

Эта потенциально мощная стратегия оптимизации может быть легко расширена для изучения приложений для оптимизации многопользовательских приложений с различными ограничениями, включая проблемы NPhard. Дальнейшие исследования могут быть сосредоточены на исследованиях чувствительности и параметров и их возможных отношениях со скоростью сходимости алгоритма. Кроме того, гибридизация с другими популярными алгоритмами, такими как PSO, также будет потенциально плодотворной. Что еще более важно, так как для большинства метаэвристических алгоритмов очень необходим математический анализ структур алгоритмов. На данный момент такой структуры нет для анализа метаэвристики вообще. Любой прогресс в этой области потенциально даст новое понимание понимания того, как и почему работают метаэвристические алгоритмы.