УДК 519.68

ИССЛЕДОВАНИЕ ЭФФЕКТИВНОСТИ ПАРАЛЛЕЛЬНОГО ГЕНЕТИЧЕСКОГО АЛГОРИТМА НА ЗАДАЧЕ КОММИВОЯЖЕРА

Семенкина О.Е.,
научный руководитель д-р техн. наук Семенкин Е.С.
Сибирский федеральный университет

Исследование эффективности стохастических алгоритмов, работающих одновременно с большим количеством текущих решений на задачах оптимизации, является актуальной научной проблемой современной математики. Данная работа посвящена исследованию эффективности распараллеленного генетического алгоритма.

Для разработки и исследования эффективности алгоритма была использована оптимизационная задача, которая часто возникает на практике, а именно задача коммивояжера. Формулировка задачи коммивояжера выглядит следующим образом:

Пусть имеется заданное множество из n городов. Требуется найти замкнутый обход минимальной длины с условием, что каждый город должен быть посещен единственный раз.

Генетический алгоритм основан на имитации естественной эволюции. Каждое решение задачи является индивидом популяции, т.е. множества решений, который представлен хромосомой – списком номеров городов в порядке их посещения. В своей работе генетический алгоритм использует несколько операторов, следующих друг за другом, таких как селекция, рекомбинация и мутация. В генетическом алгоритме для задачи коммивояжера хромосома представляет собой перестановку из n чисел (номеров городов), тогда как в стандартной версии это строка нулей и единиц. В связи с этим стандартные операторы видоизменены, однако сохраняется большое количество настраиваемых параметров, таких как вероятность мутации, тип селекции – турнирная (с выбором размера турнира), пропорциональная и ранговая (с экспоненциальным ранжированием).

Алгоритм показывает свою работоспособность и хорошую надежность при достаточном количестве ресурсов. Но необходимое количество ресурсов, а, следовательно, и время работы алгоритма, быстро растет с увеличением размерности задачи. Для решения этой проблемы используется распараллеливание алгоритмов.

Распараллеливание также применяется для снижения преждевременной сходимости к локальному оптимуму, стимуляции разнообразия и поиска альтернативных решений той же проблемы.

Существует несколько видов распараллеливания эволюционных алгоритмов:

1. Глобальные однопопуляционные

В этом случае алгоритм имеет одну главную популяцию, с которой и работают все операторы, а вычисление целевой функции распределено на несколько процессоров.

2. Однопопуляционные

В таких алгоритмах имеется одна пространственно-распределенная популяция, а операторы имеют некоторые ограничения.

3. Многопопуляционные

Здесь работа ведется с несколькими независимыми популяциями, которые периодически обмениваются индивидами, то есть осуществляется миграция, которая управляется несколькими параметрами. Многопопуляционные ГА достаточно трудны для понимания, так как эффекты миграции остаются исследованными не полностью.

При реализации параллельного алгоритма была выбрана именно многопопуляционная версия, так как она является наиболее распространенной и

интересной для изучения. К тому же вычисление пригодности индивидов в задаче коммивояжера не требует значительного количества вычислительных ресурсов.

В разработанной программе на каждом ядре компьютера развивается собственная независимая популяция, которая обменивается с остальными лучшими индивидами через определенное количество поколений. Топология их взаимодействия имеет вид клики («каждый с каждым»).

С помощью разработанной программы параллельного генетического алгоритма на тестовой задаче (решетка 5 на 5 городов) была исследована его надежность при различных настройках параметров и при различном количестве ядер, а соответственно и популяций.

В таблицах 1, 2, 3 представлена надежность алгоритма, усредненная по 100 прогонам для одного, двух и четырех ядер соответственно. Интервал между обменом индивидов был установлен на 10 поколений и с каждого ядра мигрировали по одному лучшему индивиду. В верхней строке таблицы указано количество индивидов на количество поколений. При этом в случае нескольких ядер имеется ввиду суммарное количество индивидов на всех ядрах.

Из этих таблиц можно сделать вывод, что при увеличении количества ядер, то есть отдельных популяций, качественно работа генетического алгоритма не изменяется, хотя и наблюдается некоторое изменение надежности для отдельных настроек. В то же время, существенно повышается скорость работы. Так при одном ядре один прогон занимает около 30 секунд, при двух ядрах это занимает около 16 секунд, а при четырех – около 8. То есть распараллеливание ГА позволяет существенно сократить время выполнения программы, не снижая надежности. При этом наибольшую надежность при одинаковом количестве ресурсов алгоритм показывает при низкой мутации и ранговой селекции с параметром λ=0.8 или турнирной селекцией с размером турнира k=8.

Таблица 1

Селекция		мутация	50x1600	100x800	200x400	400x200	800x100
Пропорциональная		низкая	0.64	0.44	0.42	0.40	0.38
		средняя	0.67	0.56	0.41	0.35	0.49
		высокая	0	0	0	0	0
Турнирная	k=2	низкая	0	0	0	0	0
		средняя	0	0	0	0	0
		высокая	0	0	0	0	0
	k=4	низкая	0	0.04	0.11	0.15	0.10
		средняя	0	0	0	0	0
		высокая	0	0	0	0	0
	k=8	низкая	0.98	0.65	0.50	0.73	0.80
		средняя	0.01	0.07	0.21	0.36	0.36
		высокая	0	0	0	0	0
Ранговая	λ=0.95	низкая	0	0.06	0.89	0.53	0.59
		средняя	0	0	0.28	0.74	0.70
		высокая	0	0	0	0	0
	λ=0.8	низкая	0.92	0.60	0.40	0.50	0.47
		средняя	0.02	0.90	0.52	0.55	0.55
		высокая	0	0	0	0.01	0.31
	λ=0.5	низкая	0.76	0.42	0.45	0.45	0.45
		средняя	0.77	0.65	0.44	0.48	0.48
		высокая	0	0	0.01	0.27	0.45

Таблица 2

Селекция		мутация	50x1600	100x800	200x400	400x200	800x100
Пропорциональная		низкая	0.45	0.41	0.40	0.44	0.55
		средняя	0.45	0.56	0.48	0.46	0.39
		высокая	0	0.02	0.03	0.01	0.01
Турнирная	*k*=2	низкая	0	0	0	0	0
		средняя	0	0	0	0	0
		высокая	0	0	0	0	0
	k=4	низкая	0.37	0.32	0.43	0.36	0.08
		средняя	0	0	0	0	0
		высокая	0	0	0	0	0
	k=8	низкая	0.55	0.57	0.65	0.74	0.90
		средняя	0.44	0.40	0.50	0.56	0.41
		высокая	0	0	0	0	0
Ранговая	λ=0.95	низкая	0	0	0.54	0.74	0.80
		средняя	0	0	0	0.62	0.78
		высокая	0	0	0	0	0
	λ=0.8	низкая	0.47	0.71	0.62	0.62	0.69
		средняя	0.03	0.49	0.76	0.72	0.68
		высокая	0	0	0	0	0.18
	λ=0.5	низкая	0.55	0.55	0.48	0.50	0.57
		средняя	0.40	0.61	0.59	0.52	0.51
		высокая	0	0	0.02	0.15	0.30

Таблица 3

Селекция		мутация	50x1600	100x800	200x400	400x200	800x100
Пропорциональная		низкая	0.36	0.46	0.48	0.51	0.57
		средняя	0.28	0.42	0.46	0.54	0.59
		высокая	0.01	0	0	0	0
Турнирная	*k*=2	низкая	0.08	0	0	0	0
		средняя	0	0	0	0	0
		высокая	0	0	0	0	0
	k=4	низкая	0.32	0.51	0.50	0.46	0.25
		средняя	0.10	0.03	0.02	0.01	0
		высокая	0	0	0	0	0
	k=8	низкая	0.39	0.56	0.60	0.77	0.92
		средняя	0.23	0.37	0.55	0.63	0.49
		высокая	0	0	0	0	0
Ранговая	λ=0.95	низкая	0.01	0	0.08	0.59	0.87
		средняя	0	0	0	0.04	0.62
		высокая	0	0	0	0	0
	λ=0.8	низкая	0.19	0.45	0.62	0.54	0.80
		средняя	0	0.14	0.57	0.66	0.70
		высокая	0	0	0	0	0.02
	λ=0.5	низкая	0.41	0.56	0.49	0.62	0.64
		средняя	0.21	0.46	0.63	0.62	0.68
		высокая	0	0	0	0.04	0.19

В дальнейшем предполагается разработать более эффективные методы распараллеливания генетических алгоритмов, позволяющие ускорить их работу за счет эффективного взаимодействия разнородных популяций, развивающихся на разных ядрах.