Кондратенков Валерий КМБ-19-1

**Отчет**

**Постановка задачи:** Разработать централизованный алгоритм балансировки распределенного приложения, которое представляет собой набор взаимодействующих процессов. Процессы располагаются на разных вычислительных узлах (в рамках учебной программы будут использоваться виртуальные узлы). Решение о переносе объекта с одного вычислительного узла распределенной системы на другой выполняется одним из процессов, который предварительно получает сообщения от всех вычислительных узлов об их загрузке. Сеть имеет древовидную топологию.

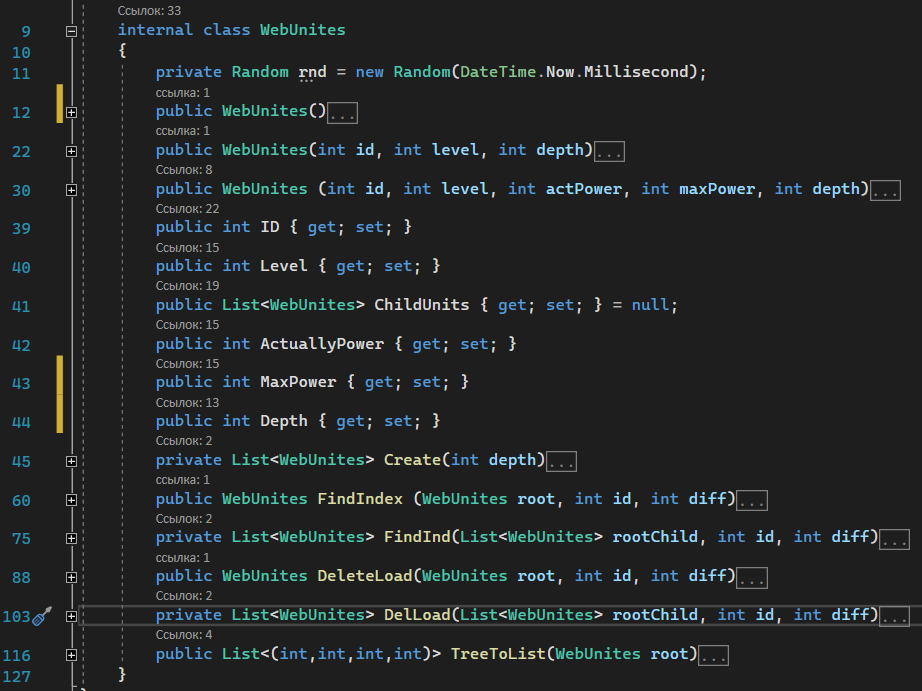
Предлагается ознакомится с волновыми алгоритмами для произведения балансировки.

**Алгоритм «Эхо»**

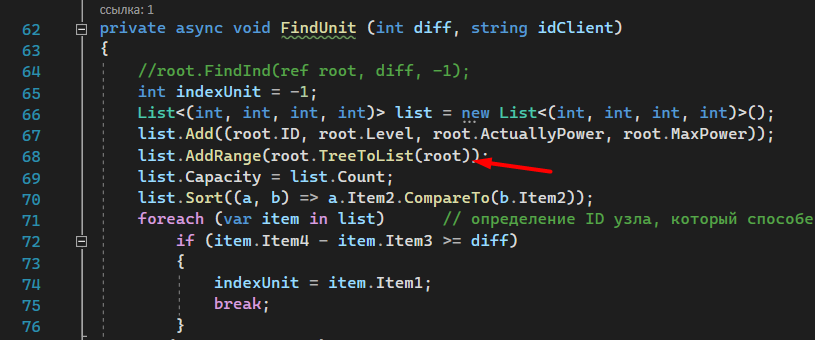
Алгоритм Эха является централизованным алгоритмом для сетей с произвольной топологией. Алгоритм наводняет сообщениями tok все процессы, выделяя таким образом остовное дерево. Маркеры возвращаются «эхом» обратно по ребрам этого дерева, напоминая поток сообщений в древесном алгоритме. Инициатор отправляет сообщение всем своим соседям. После получения первого сообщения всякий неинициатор переправляет сообщения всем своим соседям, за исключением того, от которого было получено это сообщение. Как только неинициатор получит сообщения от всех своих соседей, он отправляет эхо родительскому процессу. Как только инициатор получит сообщения от всех своих соседей, он принимает решение.

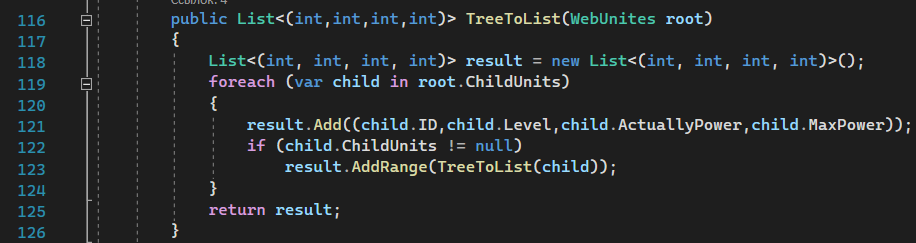
Перейдем непосредственно к реализованной программе. Она представляет из себя 2 логически различающиеся программы, связанные с помощью очередей сообщений MSMQ, – Клиент и Сервер (центральный узел и распределенные узлы в сети). Клиент может запрашивать выполнение операции определенной сложности и в зависимости от того, может сервер выполнить данную операцию или нет, формируется ответ клиенту. Каждая операция сложности P(Q) рассматривается, как единая и неделимая.

На сервере реализован класс *WebUnites*:

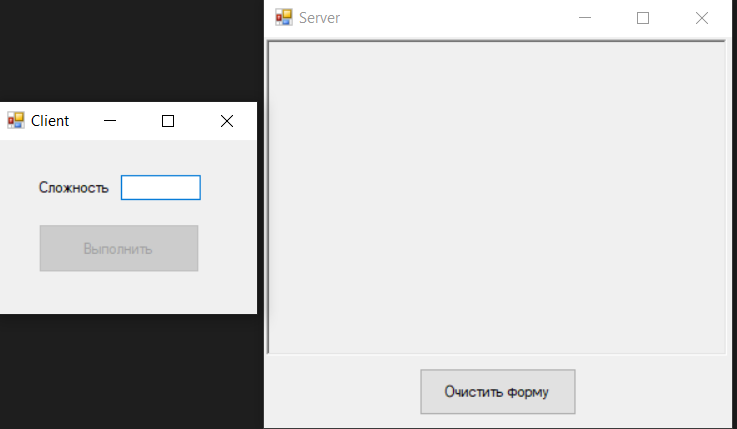


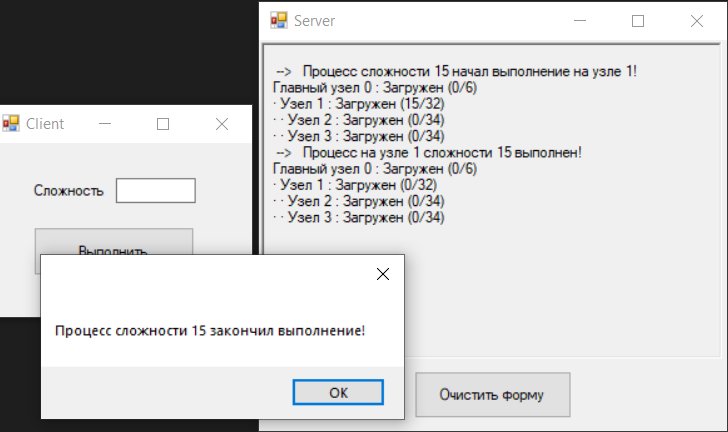
Метод *TreeToList* представляет собой обход дерева, которое было построено и использовано в качестве топологии сети, и включает использование алгоритма «Эхо», чтобы собрать актуальную информацию о загруженности каждого узла в рамках использованной топологии сети.

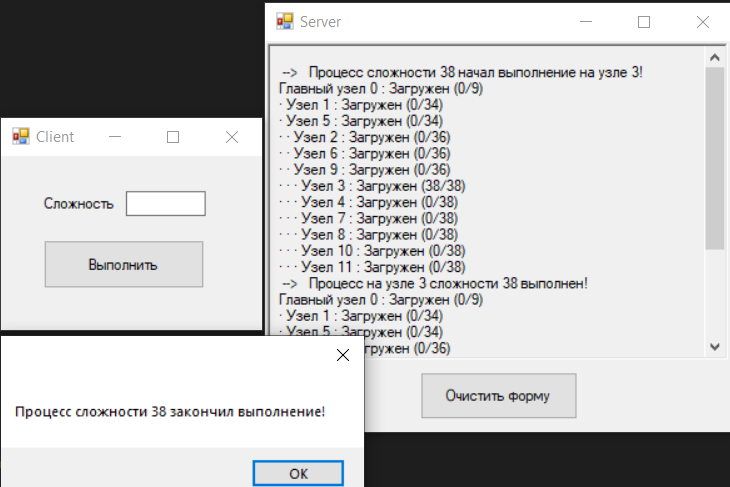


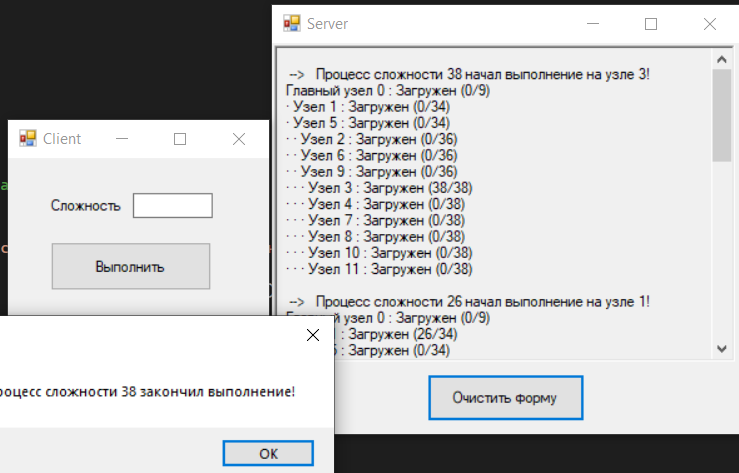


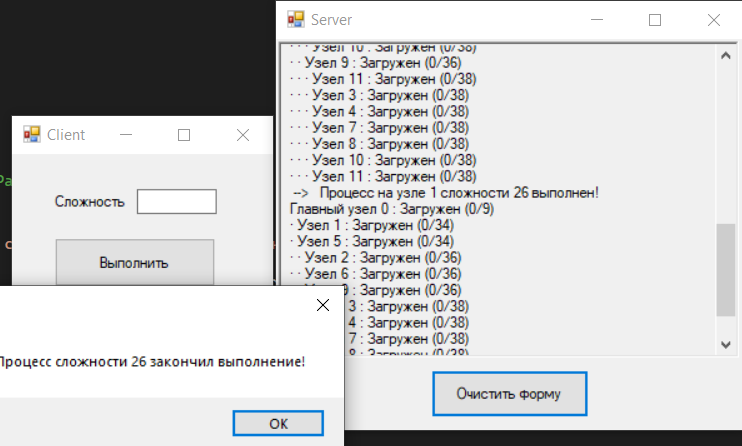
Демонстрация работы:









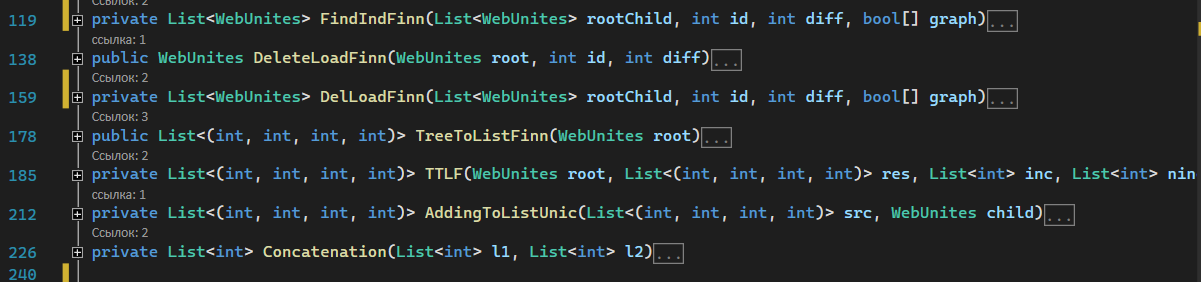
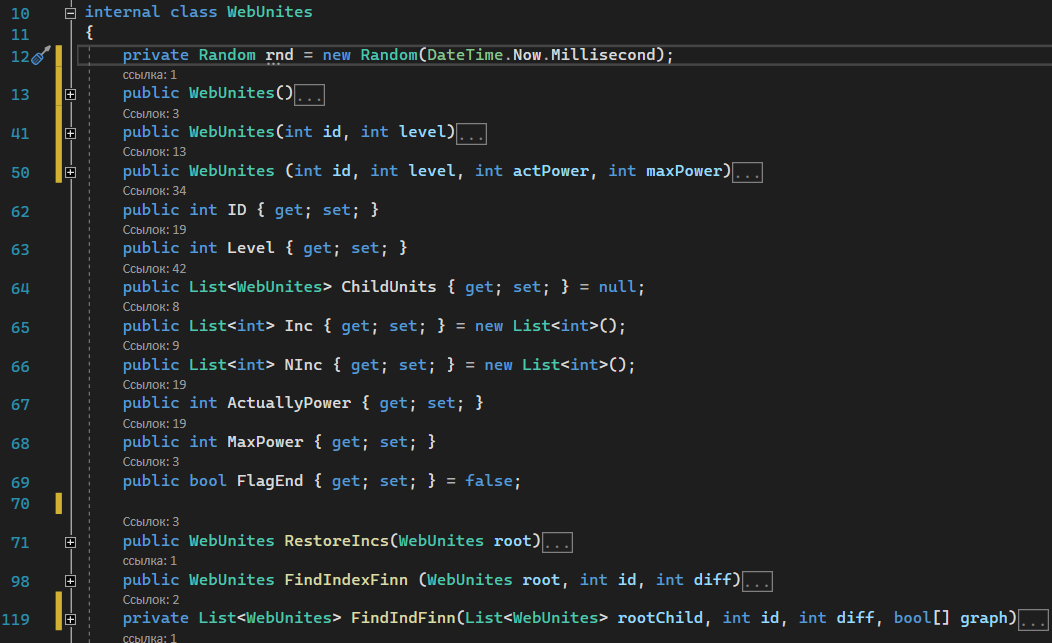


**Алгоритм «Финна»**

Алгоритм Финна — это еще один волновой алгоритм, который можно использовать для произвольных ориентированных сетей. Здесь не требуется, чтобы диаметр сети был известен заранее, но зато этот алгоритм опирается на однозначную идентифицируемость процессов. В сообщениях процессы обмениваются отличительными признаками, и это приводит к тому, что битовая сложность алгоритма становится достаточно большой. В процессе р формируются два множества отличительных признаков Incp, и Nincp. Если говорить неформально, то Incp обозначает множество таких процессов q, что некоторое событие в q предшествует самому последнему событию, случившемуся в р, a Nlncp обозначает множество таких процессов q, что у каждого соседа г процесса q какое-нибудь событие в г предшествует самому последнему событию, случившемуся в р. Вначале Incp = {р} и NIncp — 0. Процесс р отправляет сообщения, в которых содержатся Incp и Nlncp, всякий раз, когда одно из этих множеств расширяется. Когда р получает сообщение, содержащее множество Incp и множество NIncp, полученные отличительные признаки добавляются к соответствующим множествам, которые формирует процесс р. Если процесс р получил сообщения от всех своих соседей на входе, то отличительный признак р вставляется в Nlncp. Когда эти два множества станут одинаковыми, р принимает решение.

Перейдем непосредственно к реализованной программе. Она представляет из себя 2 логически различающиеся программы, связанные с помощью очередей сообщений MSMQ, – Клиент и Сервер (центральный узел и распределенные узлы в сети). Клиент может запрашивать выполнение операции определенной сложности и в зависимости от того, может сервер выполнить данную операцию или нет, формируется ответ клиенту. Каждая операция сложности P(Q) рассматривается, как единая и неделимая.

На сервере реализован класс *WebUnites*:



Метод *TreeToListFinn* представляет собой обход ориентированного графа, который был построен и использован в качестве топологии сети, и включает использование алгоритма «Финна», чтобы собрать актуальную информацию о загруженности каждого узла в рамках использованной топологии сети.

