УДК 004.045;167.1
DOI 10.18413/2411-3808-2019-46-2-311-325

ОПТИМИЗАЦИЯ СИСТЕМНО-ОБЪЕКТНЫХ ИМИТАЦИОННЫХ МОДЕЛЕЙ. ЧАСТЬ 1

OPTIMIZATION OF THE SYSTEM-THE OBJECT OF SIMULATION MODELS. PART 1

А.Г. Жихарев[1], А.А. Бузов[2], И.А. Егоров[1], А.В. Кузнецов[1], Ю.В. Жинкина[1]
A.G. Zhikharev[1], A.A. Buzov[2], I.A. Egorov[1], A.V. Kuznetsov[1], Yu.V. Zhinkina[1]

[1)] Белгородский государственный национальный исследовательский университет,
Россия, 308015, г. Белгород, ул. Победы, д. 85
[2)] АО «ВладМиВа»,
Россия, 308023, г. Белгород, ул.Садовая, д. 118

[1)] Belgorod National Research University,
85 Pobeda St, Belgorod, 308015, Russia
[2)] JSC «VladMiVa»,
18 Sadovaya St., Belgorod, 308023, Russia

E-mail: zhikharev.bsu.edu.ru

Аннотация
Системно-объектное имитационное моделирование является перспективным направлением развития системно-объектного подхода «Узел-Функция-Объект». С целью повышения эффективности системно-объектного анализа организационно-деловых и производственно-технологических процессов за счет совершенствования теоретических и инструментальных средств оптимизации системно-объектных имитационных моделей авторами выявлены несколько принципов оптимизации, которые рассматриваются в статье.
При построении имитационных моделей авторы руководствуются в том числе обобщенной характеристикой системы – мерой системности. В данной статье предложена оптимизация системно-объектной имитационной модели на примере организационно-управленческой модели «руководитель – подчиненные» в среде «UFOModeler». Обосновывается, что сформулированные принципы оптимизации позволяют перейти к разработке методов оптимизации структурных, функциональных и объектных параметров сходных моделируемых систем. Также следует отметить, что рассматриваемые в работе принципы оптимизации системно-объектных имитационных моделей не являются исчерпывающими, так как не позволяют привести оптимизируемую модель в соответствие всем общесистемным закономерностям и принципам.

Abstract
System-object simulation modeling is a promising direction for the development of the system-object approach «Node-Function-Object». In order to improve the efficiency of the system-object analysis of organizational-business and production-technological processes by improving the theoretical and instrumental tools for optimizing system-object simulation models, the authors identified several optimization principles that are considered in the article.
When building simulation models, the authors are guided, among other things, by a generalized characteristic of the system - a measure of consistency. This article proposes the optimization of the system-object simulation model on the example of the manager-subordinate organizational and managerial model in the «UFOModeler» environment. It is substantiated that the formulated principles of optimization allow us to proceed to the development of methods for optimizing the structural, functional, and object parameters of similar simulated systems. Also, it should be noted that the principles of optimization of system-object simulation models considered in the work are not exhaustive, since they do not allow the optimized model to be brought into line with all system-wide laws and principles.

Ключевые слова: принципы, имитационное моделирование, объект, система, структура, оптимизация, функция.
Keywords: principles, simulation, object, system, structure, optimization, function.

С позиции научного анализа рациональность как определенный алгоритм действия представляет собой достаточно сложное понятие. Рассматривая рациональность в качестве регулятивного принципа, мы с необходимостью учитываем его при построении системно-объектных имитационных моделей. Иными словами, оптимизация моделей рассматривается нами, с одной стороны, как процесс, способствующий построению рациональной системы; с другой стороны, как способ повышения эффективности рассматриваемой системы через максимизацию выгодных характеристик и минимизацию расходов [Маторин и др., 2018; Zhikharev and others, 2015].

С учетом вышесказанного, а также результатов, полученных в ходе рассмотрения аспектов оптимизации [Жихарев и др., 2015; Егоров и др., 2018], были сформулированы следующие принципы оптимизации системно-объектных имитационных моделей:

– принцип структурного многообразия (системности);
– принцип ограниченной рациональности (в теориях Нельсона, Уинтера);
– принцип воспроизведения рациональности (максимизации);
– принцип оптимизации (эффективного распределения ресурсов).

Внутренняя причина наличия существенных функциональных связей (потоков) системы обуславливает саму сущность системы. Вместе с тем мера системы определяется различными способами:

1) топологической энтропией как сложностью конфигурации структуры (в физике);

2) числом операций или операндов, используемых для конечного результата в рамках допустимого входного набора (в формальных системах, имитационных моделях).

Для раскрытия принципа структурного многообразия через выявление специфики упорядоченности воспользуемся дефиницией обобщенной характеристики системы – меры системности. При рассмотрении формальных, имитационных систем (моделей) мера системности определяется отношением множества возможных функциональных состояний к множеству требуемых функциональных состояний потокового объекта [Казиев, 2007]. В данном смысле, согласно принципу структурной упорядоченности, оптимизация модели происходит через создание таких связей между объектами подсистемы, при которых мера системности будет равна или близка к единице [Жихарев и др., 2014; Zhikharev and, 2018]. Таким образом, мера системы определяется нами на числовом промежутке от нуля до единицы.

В качестве примера учета принципа структурной упорядоченности рассмотрим организационно-управленческую модель «руководитель – подчиненные» в среде UFOModeler (Рис. 1).

Рис. 1. Модель «руководитель – подчиненные»
Fig. 1. The model of «leader – subordinates»

Руководитель отдела программирования имеет определенный список типов заданий. Каждому типу соответствует уникальный номер. Руководитель ставит задания подчиненным согласно типу заданий. Программисты последовательно выполняют поступающие задания, в случае успешного выполнения очередного задания они докладывают о завершении руководителю. При этом руководитель выдает новое задание на выполнение.

Рассмотрим структуру потокового объекта «Задание» [Zhikharev et al., 2016; Маторин и др., 2018] в упомянутой выше модели, от которого происходит наследование двух потоковых объектов «Задание 1» и «Задание 2» (рис. 2).

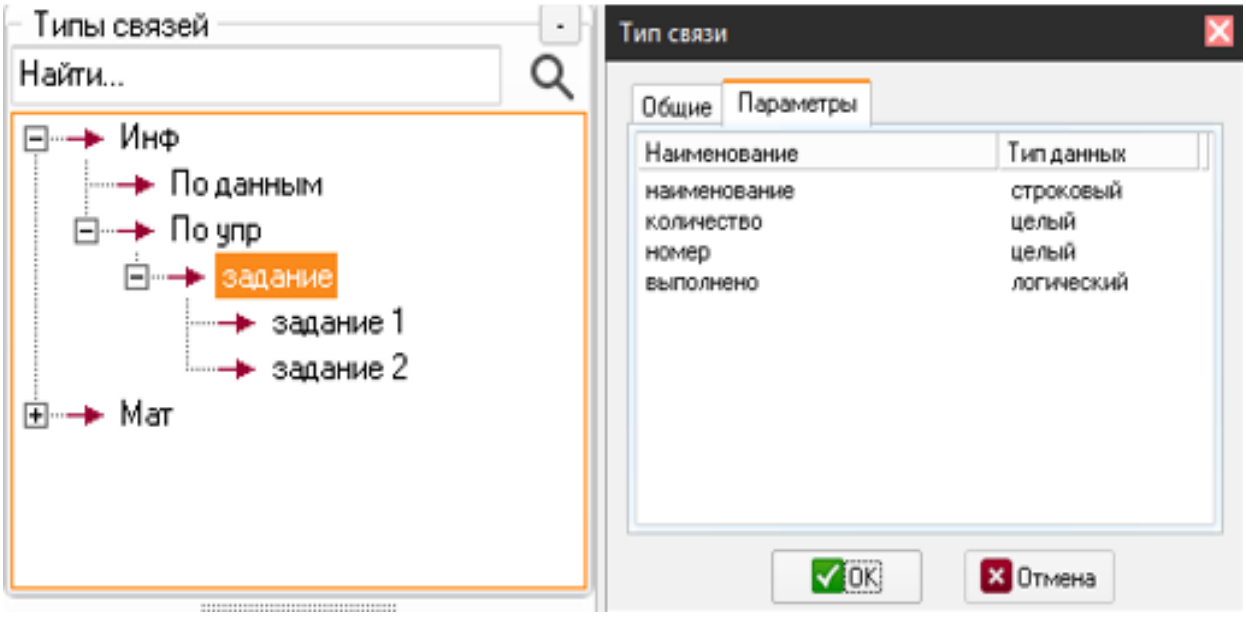

Рис. 2. Иерархия связей и параметры потокового объекта «задание»
Fig. 2. The hierarchy of relations and parameters of the stream object «task»

В данных потоковых объектах выявлены четыре параметра:

– «наименование» – строковый параметр, описание выполняемого в данный момент типа задания;

– «номер» – целочисленный тип, численное представление выполняемого в данный момент типа задания;

– «количество» – целочисленный параметр, подсчет выполненных программистами заданий;

– «выполнено» – логический тип, передача сигнала руководителю от исполнителей о завершении выполнения очередного задания.

Узел «руководитель отдела программирования» соединен соответственно с узлами «инженер-программист 2 категории» и «ведущий инженер» потоковыми объектами «Задание 1» и «Задание 2».

Примем список типов заданий узла «руководитель отдела программирования» за количество требуемых функциональных состояний [Matorin and others, 2018; Маторин и др. 2013], в которых могут находиться потоковые объекты «Задание 1» и «Задание 2». Согласно выполняемому в данный момент типу задания руководитель посылает сигнал через параметр «номер» потоковых объектов «Задание 1» и «Задание 2».

Составим список из шести типов заданий и запишем их словесное описание в свойства объекта «генератор заданий» узла «руководитель отдела программирования»:

– «task1» – найти ошибку в программном коде;
– «task2» – исправить ошибку в программном коде;
– «task3» – проанализировать сторонний программный код;
– «task4» – доработать сторонний программный код;

– «task5» – написать новый программный метод;
– «task6» – проработать модульность программного продукта.

Приведенные выше типы заданий имеют уникальные числовые индексы для идентификации среди всех сотрудников отдела программирования; индексы объявлены в качестве констант в программном коде всех узлов.

Узлы «инженер-программист 2 категории» и «ведущий инженер» имеют список типов заданий, который они умеют выполнять, данный список формирует количество возможных функциональных состояний потоковых объектов «Задание 1» и «Задание 2». Через параметр «номер» потоковых объектов «Задание 1» и «Задание 2» поступает команда на исполнение задания. Инженер-программист умеет выполнять три типа заданий: «task1», «task2», «task3». Ведущий инженер умеет выполнять пять типов заданий: «task1», «task2», «task3», «task4», «task6». Если в блоке «инженер-программист 2 категории» и «ведущий инженер» может выполнить данный тип задания, то для исходящей связи «Задание 1» и «Задание 2» параметр «количество» увеличивается на единицу соответственно.

Программный код узла «руководитель отдела программирования» приведен ниже (рис. 3).

Скрипт

```
const
 task_count = 6; // количество требуемых заданий (состояний)

var tasknum1, tasknum2: integer;
begin
  tasknum1 := 0;  // инициализируем с условным номером задания
  tasknum2 := 2;
  while true do
  begin
    if getlinkoutB('задание 1.выполнено') then
    begin
      setlink( 'задание 1.номер', tasknum1 );
      setlink( 'задание 1.наименование', getObjPropS('task' + IntToStr(tasknum1 + 1 ) )

      tasknum1 := tasknum1 + 1;
      if tasknum1 = task_count then tasknum1 := 0;
      setlinkout('задание 1.выполнено', false);
    end;

    if getlinkoutB('задание 2.выполнено') then
    begin
      setlink( 'задание 2.номер', tasknum2 );
      setlink( 'задание 2.наименование', getObjPropS('task' + IntToStr(tasknum1 + 1) )

      tasknum2 := tasknum2 + 1;
      if tasknum2 = task_count then tasknum2 := 0;

      setlinkout('задание 2.выполнено', false);
    end;
    delay(10);
  end;
end.
```

Рис. 3. Реализация функции узла «руководитель отдела программирования»
Fig. 3. Implementing the function of the node «head of programming»

Программный код узлов «инженер-программист 2 категории» и «ведущий инженер» подобен и выглядит следующим образом (рис. 4).

Скрипт

```
const
  task1 = 0;  // единые уникальные номера заданий принятые в отделе программирования
  task2 = 1;
  task3 = 2;
var ableToDoTasks: array of integer; // массив с номерами заданий, который может выполн
taskType,                            // поставленный руководителем тип задания
i: integer;                          // счетчик для перебора элементов массива
begin
  setLength(ableToDoTasks, 3);
  ableToDoTasks[0] := task1 ;
  ableToDoTasks[1] := task2 ;
  ableToDoTasks[2] := task3 ;

  while true do
  begin
    if not getlinkinB('задание 1.выполнено') then
    begin
      taskType := getlinkinI('задание 1.номер');
      for i := 0 to High(ableToDoTasks) do
        if  ableToDoTasks[i] = taskType then
        begin
          setlinkout('задание 1.количество', getlinkoutI('задание 1.количество') + 1 );
          setobjprop( '#buzy', 100 );
          break;
        end;
      setlinkin('задание 1.выполнено', true);
    end;
    delay(10);
    setobjprop( '#buzy', 0 );
  end;
end.
```

Рис. 4. Реализация функции узла «инженер-программист 2 категории»
Fig. 4. Implementation of the function of the site «software engineer 2 categories»

Запустим модель на исполнение. Красная полоса узла «ведущий инженер» означает завершение выполнения очередного задания. Белая полоса узла «инженер-программист 2 категории» означает, что задание находится в процессе выполнения. Количество выполненных заданий на момент остановки выполнения модели у инженера-программиста 2 категории составляет 1266 заданий, количество выполненных заданий у ведущего инженера – 1953. Темп прироста выполненных заданий у ведущего инженера составляет примерно 35%, факт прироста обусловлен тем, что он предоставляет большее количество возможных функциональных состояний (рис. 5).

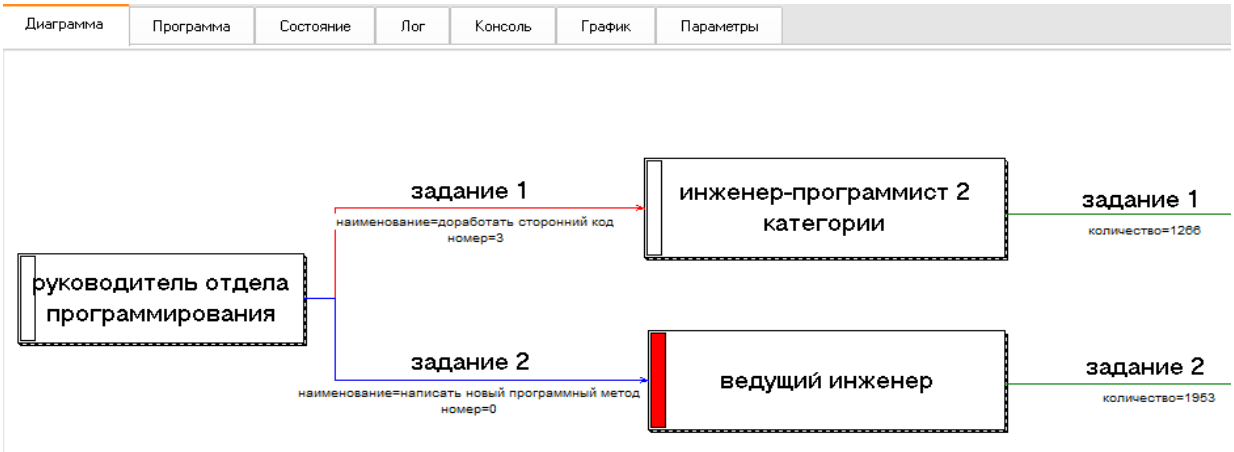

Рис. 5. Имитация функционирования системы
Fig. 5. Simulated system operation

Рассчитаем меру системности для потоковых объектов «Задание 1» и «Задание 2». Для потокового объекта «Задание 1» мера системности равна:

$$MOS1 = \frac{3}{6} = 0.5 \quad (1)$$

Для потокового объекта «Задание 2» мера системности равна:

$$MOS2 = \frac{5}{6} = 0.83 \quad (2)$$

Сравним результаты вычислений (1) и (2). Так как мера системности (2) выше, то потоковый объект «Задание 1» не в полной мере соответствует требованиям руководителя. С точки зрения оптимизации и принципа структурной упорядоченности необходимо устранить связь «Задание 1».

Таким образом, инженер-программист 2 категории не обладает необходимым уровнем компетенций. Данный сотрудник нуждается в дополнительном обучении (повышении квалификации) для обеспечения возможности выполнения поставленных заданий. В противном случае руководство компании может вынести решение об увольнении сотрудника в связи с несоответствием занимаемой должности.

Принцип алгоритмической рациональности состоит в выборе эффективного алгоритма решения из множества существующих в рамках одного УФО-объекта [Abadi, 1996; Жихарев и др., 2014].

Для алгоритмов сортировки основополагающим фактором для выбора рационального алгоритма служит его временная и пространственная сложность. При этом, как правило, временная и пространственная сложность должна выбираться в соответствии с исходным объемом данных, участвующих в обработке алгоритмом.

Например, при рассмотрении алгоритмов сортировки с исходным массивом данных, количество элементов которого не превышает ста, допускается использование простых алгоритмов перебора с временной сложностью O(n2). Если количество элементов исходного массива гораздо больше, эффективнее использовать усложненные алгоритмы с временной сложностью O(n log n) или O(n (log n)2) [Кнут, 2016].

Важным аспектом при выборе алгоритма сортировки является оценка объема дополнительной памяти, затрачиваемой на выполнение алгоритма. Так называемые рекурсивные алгоритмы сортировки используют дополнительную стековую память при упорядочивании элементов массива. Например, для алгоритма «Быстрой сортировки» требуется в среднем O(log n) дополнительной памяти, а в худшем случае O(n). Таким образом, в контексте использования памяти также прослеживается зависимость выбора алгоритма сортировки от входного количества элементов [Кнут, 2016].

Необходимо учитывать некоторые нюансы. Так, например, «Быстрая сортировка» имеет высокую вероятность к деградации до временной сложности O(n2) при неудачном выборе опорного элемента [Кнут, 2014].

Рассмотрим пример модели, в которой сортируется массив книг библиотечного фонда по уникальному индексу, присвоенному при добавлении в базу данных. Используем среду имитационного моделирования «UFOModeler» [Kondratenko and others, 2017] для построения модели. Отталкиваясь от правила, что выбор алгоритма должен зависеть от количества рассматриваемых элементов, отсортируем двумя алгоритмами сначала массив индексов, состоящий из двадцати элементов, а затем теми же алгоритмами отсортируем массив из тысячи элементов. Выберем алгоритмы, соответствующие алгоритмической сложности, указанной выше. «Пузырьковая сортировка» для сложности O(n2) и «Сортировка Шелла» для – O(n (log n)2). Особенность данных алгоритмов в том, что для них практически не требуется выделение дополнительной памяти. Дополнительная память оценивается как O(1) (рис. 6).

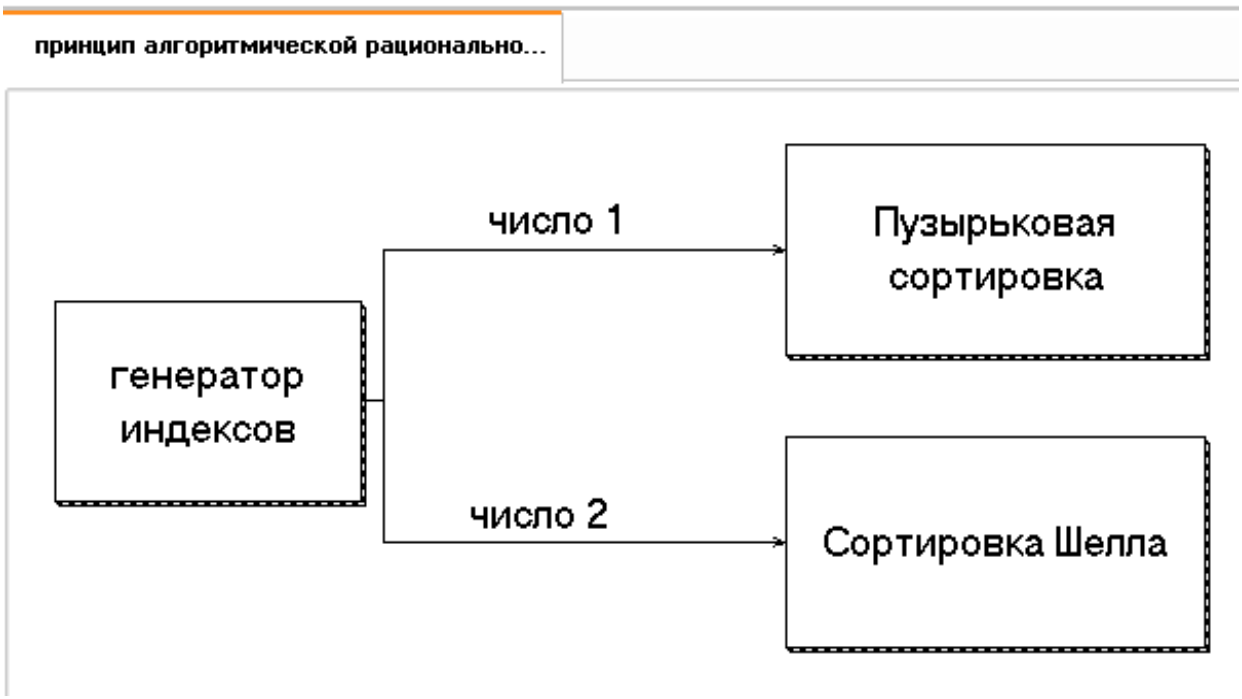

Рис. 6. Модель сортировки книг библиотечного фонда
Fig. 6. Library stock sorting model

Представим структуру потокового объекта «Число», наследниками которого являются два потоковых объекта «Число 1» и «Число 2». Потоковый объект «Число» обладает следующими параметрами:

– значение – целочисленный тип, генерирует псевдослучайное число;

– запись – логический тип, выдает сигнал об успешном добавлении очередного сгенерированного числа в неотсортированный массив;

– старт – логический тип, выдает сигнал разрешающий начать сортировку массива (рис. 7).

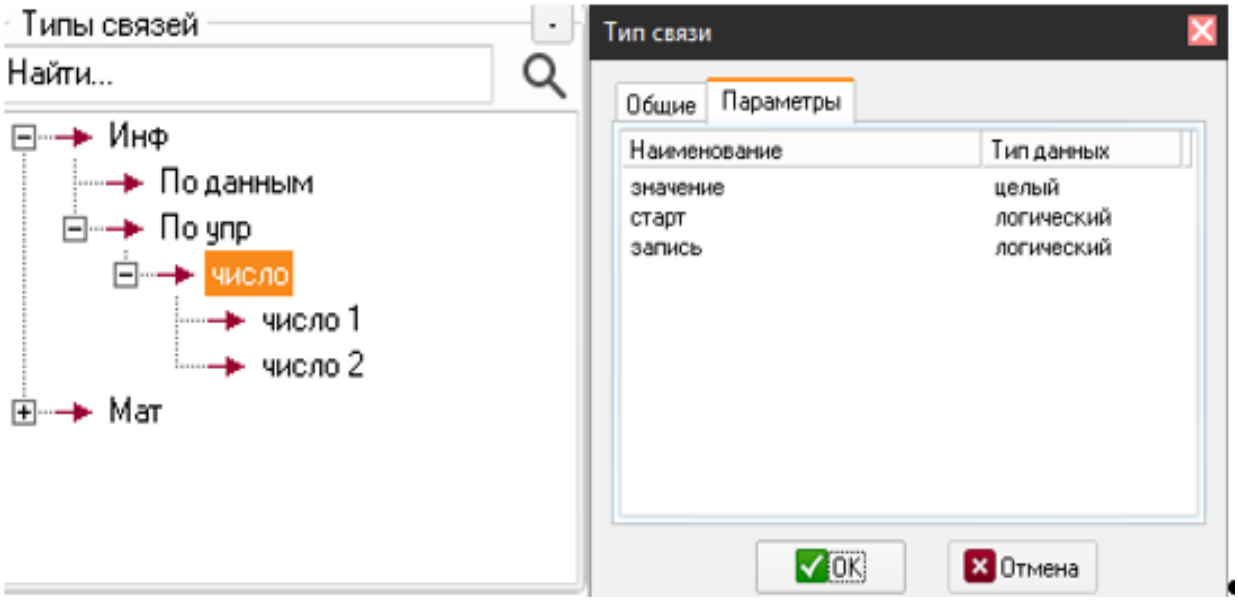

Рис. 7. Иерархия связей и параметры потокового объекта «Число»
Fig. 7. Relationship hierarchy and parameters of the «Number» stream object

Для точной оценки времени, затраченного на сортировку, узел «Генератор индексов» будет формировать одинаковые массивы псевдослучайных чисел для узлов «Пузырь-

ковая сортировка» и «Сортировка Шелла». Количество чисел для передачи записано в константе «n». Псевдослучайные числа выдаются через параметр «значение». При этом узел «Генератор индексов» ожидает сигнала через параметр «запись» от потоковых объектов «Число 1» и «Число 2» о том, что числа успешно переданы в оба узла, затем генерируется следующее число. После передачи необходимого количества индексов узел «Генератор индексов» выдает сигнал через параметр «старт» потоковых объектов «Число 1» и «Число 2» узлам для начала процесса сортировки.

Программная реализация узла «Генератор индексов» на языке УФО-скрипт [Жихарев и др., 2013] приведена ниже (рис. 8).

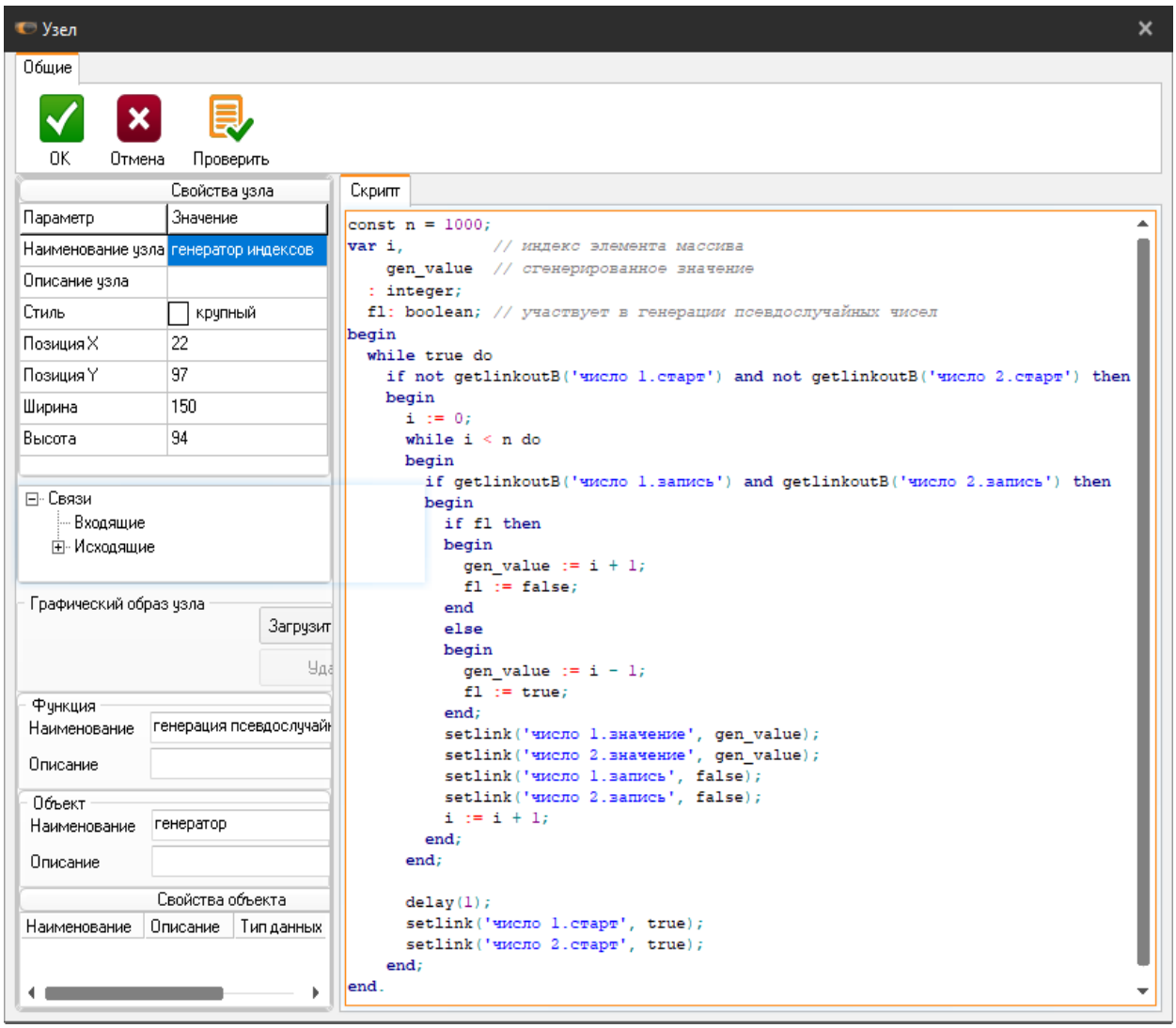

Рис. 8. Реализация функции узла «Генератор индексов»
Fig. 8. Implementing the function of the Index Generator node

Реализация узла «Пузырьковая сортировка» состоит из двух частей. Первая часть осуществляет прием сгенерированных псевдослучайных чисел и добавляет их в массив индексов «sorting», вторая часть осуществляет сортировку элементов согласно алгоритму пузырьковой сортировки [Кнут, 2014].

Программный код узла «Пузырьковая сортировка» выглядит следующим образом (рис. 9).

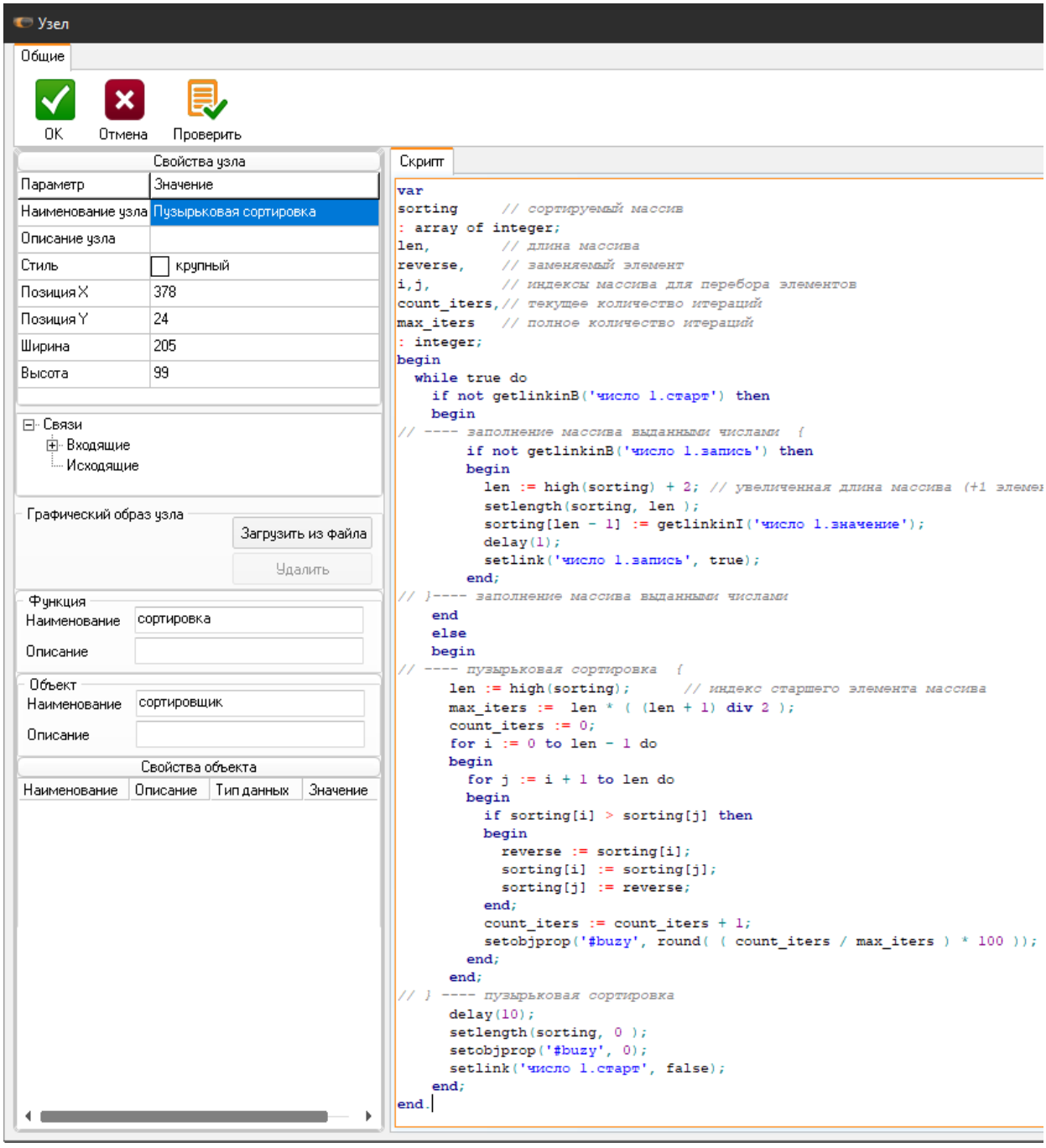

Рис. 9. Реализация функции узла «Пузырьковая сортировка»
Fig. 9. Implementing the function of the Bubble Sort node

Реализация узла «Сортировка Шелла» также подразделяется на две части. Первая часть осуществляет прием сгенерированных псевдослучайных чисел и добавляет их в массив индексов «sorting», вторая часть упорядочивает элементы согласно алгоритму сортировки Шелла [Кнут, 2014].

Программный код узла «Сортировка Шелла» выглядит следующим образом (Рис. 10).

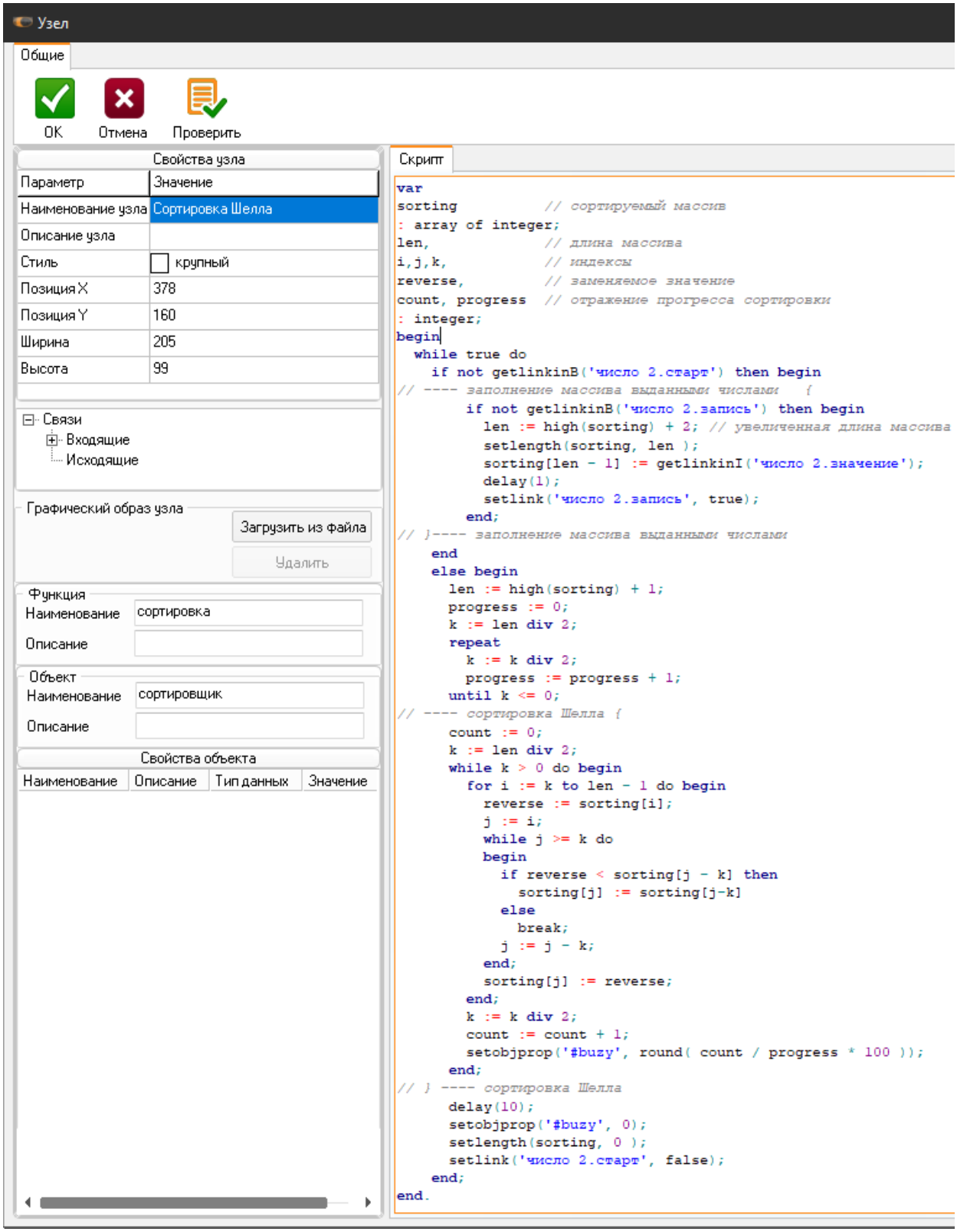

Рис. 10. Реализация функции узла «Сортировка Шелла»
Fig. 10. Implementing the function of the Shell Sort node

Зададим константу «n» в программном коде узла «Генератор индексов» равную двадцати и запустим модель на исполнение (рис. 11).

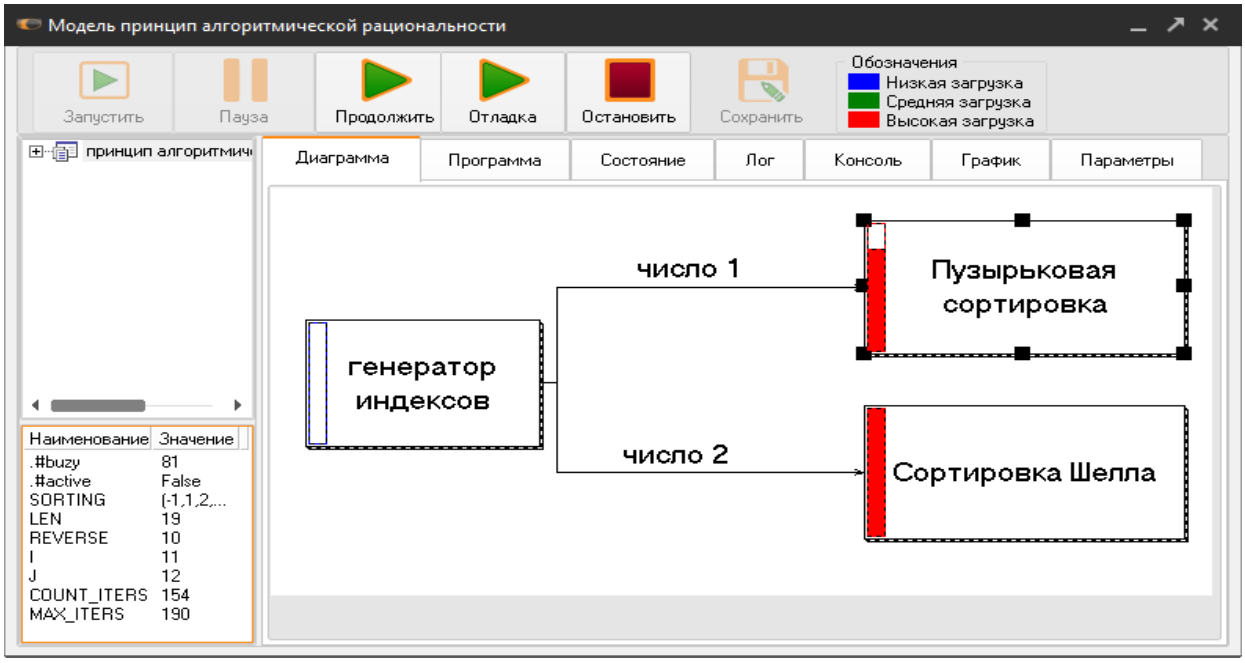

Рис. 11. Результат выполнения модели для сортировки двадцати элементов
Fig. 11. Result of running a model to sort twenty items

Не полностью закрашенная красная полоса слева от названия узла «Пузырьковая сортировка» обозначает продолжающийся процесс сортировки. Красная полоса у узла «Сортировка Шелла» говорит о том, что массив успешно отсортирован.

Несмотря на то, что процесс сортировки пузырьком был завершен лишь на 81%, сортировка методом пузырька завершила работу практически одновременно с сортировкой Шелла, так как дополнительное время в масштабе 1:1 (реальное/моделируемое) составило не более 100 мс (приблизительная цифра), что свидетельствует о допустимости использования пузырьковой сортировки для небольших массивов.

Изменим количество генерируемых чисел в узле «Генератор индексов» и запустим модель на исполнение для тысячи элементов (рис. 12).

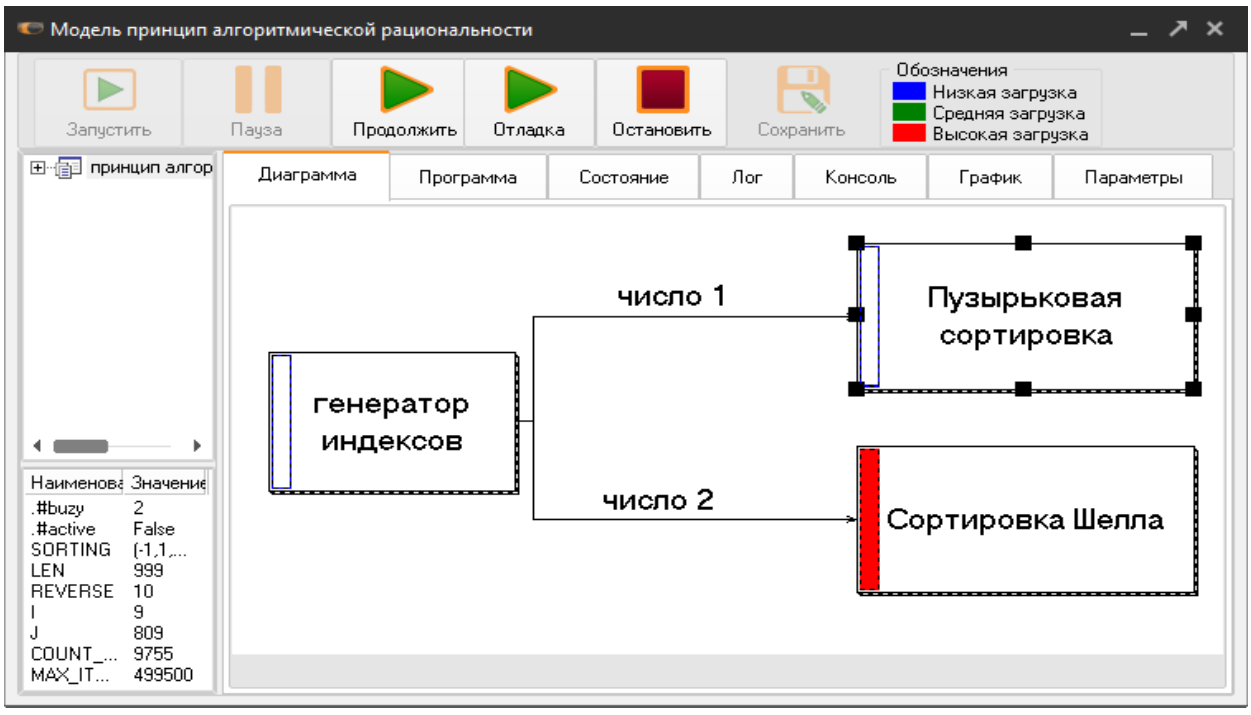

Рис. 12. Результат выполнения модели для сортировки тысячи элементов
Fig. 12. Result of running the model to sort thousands of items

В результате работы модели «Сортировка Шелла» завершила упорядочивание массива гораздо быстрее, чем «Пузырьковая сортировка». Отсюда следует, что с ростом количества элементов массива становится все более рациональным использование усложненных алгоритмов сортировок, таких как «сортировка Шелла», «Быстрая сортировка» и другие.

Таким образом, выбор алгоритма для оптимизируемой функции по принципу алгоритмической рациональности требует анализа многих факторов, окончательный результат анализа заключается в применении эффективного алгоритма относительно временной и пространственной сложности. В рамках данного принципа оптимизации планируется разработка таблицы соответствия выбора алгоритма от совокупности признаков входных данных.

При рассмотрении принципа эффективного распределения ресурсов необходимо говорить об оптимизации как о процессе максимизации целевой функции при заданных ограничениях. Принцип эффективного распределения ресурсов заключается в получении максимальной выгоды от имеющихся управленческих и производственных ресурсов для решения поставленных задач.

Раскроем принцип эффективного распределения ресурсов через создание примерной модели в среде «UFOModeler». Кондитерское предприятие получает прибыль от производства двух видов продукции, прибыль от продажи одного пирожного составляет сорок пять рублей, прибыль от продажи мороженого составляет восемьдесят рублей, потоковый объект «молоко» содержит количественный параметр, равный четыремстам упаковок, а потоковый объект «сахар» содержит количественный параметр равный четыремстам пятидесяти упаковок. Оба ингредиента одновременно применяются в обоих продуктах. В каком количестве необходимо выпустить пирожных и мороженых, чтобы получить максимальную прибыль? (рис. 13).

Рис. 13. Модель производства кондитерских изделий
Fig. 13. Model of confectionery production

Изначально модель построена так, что продукция двух видов будет выпускаться одновременно до тех пор, пока хватает ингредиентов на производство еще одной единицы продукции каждого вида. С помощью узла «симплекс-метод» найдем решение поставленной задачи для нахождения максимальной прибыли, при этом модель примет следующий вид (рис. 14).

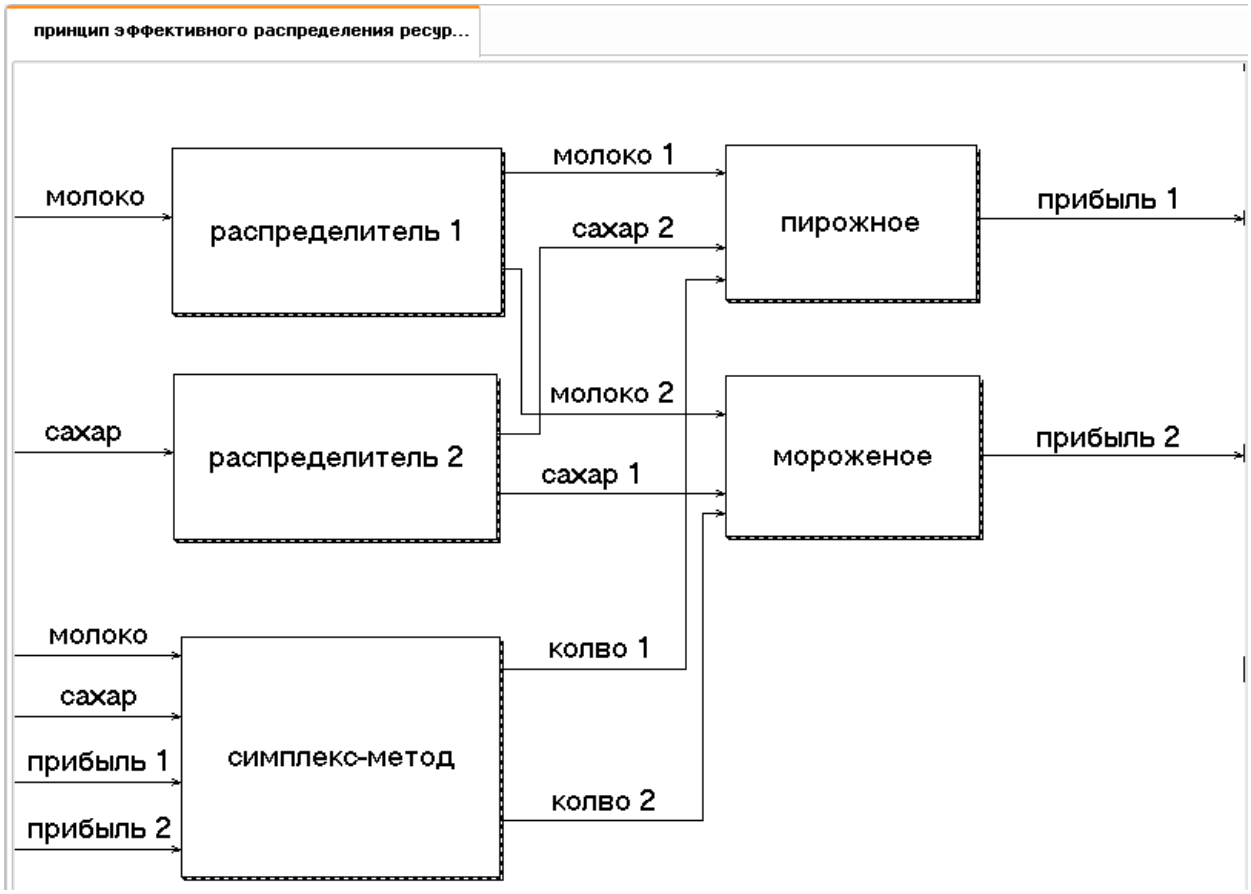

Рис. 14. Оптимизированная модель производства кондитерских изделий
Fig. 14. Optimized confectionery production model

Алгоритм узла «симплекс-метод» осуществит нахождение оптимального количества для выпуска двух видов продукции с учетом существующих ингредиентов. В данной модели пирожных необходимо выпустить в количестве двадцати четырех штук, а мороженого – четырнадцати штук, при этом оптимальные значения будут записаны в свойство объекта «необх_колво» для пирожного и мороженого. Таким образом, согласно принципу эффективного распределения ресурсов, оптимизация модели привела к увеличению прибыли предприятия до двух тысяч двести у. е.

Таким образом, сформулированные принципы оптимизации позволяют перейти к разработке методов оптимизации структурных, функциональных и объектных параметров моделируемых систем.

Благодарности

Исследования выполнены при финансовой поддержке проектов Российского фонда фундаментальных исследований №№ 18-07-00355, 19-07-00290, 19-07-00111.

Список литературы
References

1. Егоров И.А., Жихарев А.Г., Маторин С.И. 2018. К вопросу оптимизации системно-объектных имитационных моделей. Тезисы XIX Всероссийской конференции молодых ученых по математическому моделированию и информационным технологиям, г. Кемерово, 29 октября – 02 ноября, 61–62.

Egorov I.A., Zhiharev A.G., Matorin S.I. 2018. K voprosu optimizacii sistemno-ob"ektnyh imitacionnyh modelej [On the issue of optimizing system-object simulation models]. Tezisy XIX Vserossijskoj konferencii molodyh uchenyh po matematicheskomu modelirovaniyu i informacionnym tekhnologiyam, g. Kemerovo, 29 oktyabrya – 02 noyabrya [Abstracts of the XIX All-Russian Conference

of Young Scientists on Mathematical Modeling and Information Technologies, Kemerovo, October 29-November 2], 61–62.

2. Жихарев А.Г., Болгова Е.В., Гурьянова И.В., Маматова О.П. 2014. О перспективах развития системно-объектного метода представления организационных знаний. Научные ведомости Белгородского государственного университета. Сер. История. Политология. Экономика. Информатика.1(172): 110–114.

Zhiharev A.G., Bolgova E.V., Gur'yanova I.V., Mamatova O.P. 2014. About prospects of development of the system and object method of representation of organizational knowledge. Belgorod State University Scientific Bulletin. History. Political science. Economics. Information technologies. 1(172): 110–114.

3. Жихарев А.Г., Корчагина К.В., Бузов П.А., Акулов Ю.В., Жихарева М.С. 2016. Об имитационном моделировании производственно-технологических систем. Сетевой журнал «Научный результат», серия «Информационные технологии». 1(3): 25–31.

Zhiharev A.G., Korchagina K.V., Buzov P.A., Akulov Yu.V., Zhihareva M.S. 2016. About simulation modeling of production and technological systems. Research result. Information technologies. 1(3): 25–31.

4. Жихарев А.Г., Маторин С.И. 2014. Системно-объектное моделирование технологических процессов. Научные ведомости БелГУ. Сер. История. Политология. Экономика. Информатика. 21(192): 137–141.

Zhiharev A.G., Matorin S.I. 2014. System-object modeling of technological processes. Belgorod State University Scientific Bulletin. History. Political science. Economics. Information technologies. 21(192): 137–141.

5. Жихарев А.Г., Маторин С.И., Зайцева Н.О. 2015. Системно-объектное имитационное моделирование транспортных и технологических процессов. Научные ведомости БелГУ. Сер. История. Политология. Экономика. Информатика. 7(204): 159–169.

Zhiharev A.G., Matorin S.I., Zajceva N.O. 2015. System-object simulation modeling of transport and technological processes. Belgorod State University Scientific Bulletin. History. Political science. Economics. Information technologies. 7(204): 159–169.

6. Жихарев А.Г., Маторин С.И., Зайцева Н.О. 2015. Системно-объектный инструментарий для имитационного моделирования технологических процессов и транспортных потоков. Искусственный интеллект и принятие решений. 4: 95–103.

Zhiharev A.G., Matorin S.I., Zajceva N.O. 2015. System-object tools for simulation modeling of technological processes and transport flows. Artificial Intelligence and Decision Making. 4: 95–103.

7. Жихарев А.Г., Маторин С.И., Маматов Е.М., Смородина Н.Н. 2013. О системно-объектном методе представления организационных знаний. Научные ведомости БелГУ. Сер. История. Политология. Экономика. Информатика. 8(151): 137–146.

Zhiharev A.G., Matorin S.I., Mamatov E.M., Smorodina N.N. 2013. On the system-object method of presenting organizational knowledge. Belgorod State University Scientific Bulletin. History. Political science. Economics. Information technologies. 8(151): 137–146.

8. Казиев В.М. 2007. Введение в анализ, синтез и моделирование систем. 2-е издание. М., Бином. Лаборатория знаний Серия: Основы информационных технологий.

Kaziev V.M. 2007. Vvedenie v analiz, sintez i modelirovanie system [Introduction to the analysis, synthesis and modeling systems]. 2-e izdanie. M., Binom. Laboratoriya znanij Seriya: Osnovy informacionnyh tekhnologij.

9. Кнут Д.Э. 2016. Искусство программирования. Т. 1. Основные алгоритмы. М., Вильямс, 720.

Knut D.E. 2016. Iskusstvo programmirovaniya. T. 1. Osnovnye algoritmy [The art of programming. T. 1. Basic algorithms]. M., Vil'yams, 720.

10. Кнут, Д.Э. 2014. Искусство программирования. Т. 3. Сортировка и поиск. М., Вильямс, 832.

Knut, D.E. 2014. Iskusstvo programmirovaniya. T. 3. Sortirovka i poisk [The art of programming. V. 3. Sorting and search]. M., Vil'yams, 832.

11. Маторин С.И., Жихарев А.Г, Зайцева Н.О., Брусенская И.Н. 2013. Имитационное моделирование транспортных потоков с применением УФО-подхода. Научные ведомости БелГУ. Сер. История. Политология. Экономика. Информатика. 22(165): 148–153.

Matorin S.I., Zhiharev A.G, Zajceva N.O., Brusenskaya I.N. 2013. Simulation modeling of traffic flows using the UFO approach. Belgorod State University Scientific Bulletin. History. Political science. Economics. Information technologies. 22(165): 148–153.

12. Маторин С.И., Жихарев А.Г. 2018. Общесистемные закономерности как содержательные элементы системной теории, основанной на системно-объектном подходе. Научные ведомости БелГУ. Серия: Экономика. Информатика, 45(2): 372-385.

Matorin S.I., Zhiharev A.G. 2018. System-wide regularities as meaningful elements of a system theory based on a system-object approach. Belgorod State University Scientific Bulletin. Economics. Information technologies, 45(2): 372–385.

13. Маторин С.И., Жихарев А.Г. 2018. Формализация системно-объектного подхода «Узел-Функция-Объект». Прикладная информатика, том. 13, 3(75): 124–135.

Matorin S.I., Zhiharev A.G. 2018. Formalization of the system-object approach «Unit-Function-Object». Journal of Applied Informatics. 13, 3(75): 124–135.

14. Маторин С.И., Жихарев А.Г., Зимовец О.А. 2016. Системно-объектное моделирование адаптации эволюции экономических систем. Вестник Белгородского университета кооперации, экономики и права. 4(60): 81–92.

Matorin S.I., Zhiharev A.G., Zimovec O.A. 2016. System-object modeling adaptation of the evolution of economic systems. Bulletin of Belgorod University of Cooperation, Economics and Law. 4(60): 81–92.

15. Abadi M, Cardelli L. 1996. A Theory of Objects, Springer, Verlag.

16. Kondratenko A.A., Matorin S.I., Zhikharev A.G., Nemtsev A.N., Riabtceva I.N. 2017. Application of logical output means on ontologies to UFO models of subject domains. Journal of Engineering and Applied Sciences. 12(5): 1347–1354.

17. Matorin S.I., Zhikharev A.G. 2018. Calculation of the function objects as the systems formal theory basis. Advances in Intelligent Systems and Computing, 679: 182–191.

18. Zhikharev A., Matorin S., Egorov I. 2018. Formal principles of system-object simulation modeling of technological and production processes. Journal of Advanced Research in Dynamical and Control Systems, 10(10 Special Issue): 1806–1812.

19. Zhikharev A.G., Matorin S.I., Kuznetsov A.V., Zherebtsov S.V., Tchekanov N.A. 2018. To The Problem of the Coefficient Calculus of the Nodal Object in the System-Object Models. Jour of Adv Research in Dynamical & Control Systems. 10(10-Special Issue): 1813–1817.

20. Zhikharev A.G., Matorin S.I., Zaitseva N.O. 2015. About perspectives of simulation technological processes functioning with using system-object approach node-function-object. International Journal of Applied Engineering Research. 10(12): 31363–31370.

Zhikharev A.G., Matorin S.I., Zimovets O.A., Zhikhareva M.S., Rakov V.I. 2016. The simulation modeling of systems taking into account their internal parameters change. International Journal of Pharmacy & Technology. 8(4): 26933–26945.