электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

УДК 00

Разработка методики календарного планирования производственных процессов на базе методологии бережливого производства

 # 09, сентябрь 2012

Мухлисуллина Э.Т.

Научный руководитель Ю.В. Берчун
МГТУ им. Н.Э.Баумана, Москва, Российская Федерация

МГТУ им. Н.Э. Баумана
bauman@bmstu.ru

1. Формализация постановки задачи

Рассмотрим формальную постановку задачи календарного планирования производства. Пусть O – множество заказов, J – множество работ, I – множество станков, K – множество стадий, объединяющих однотипные станки, так что $N = \sum_{j \in J} k_j$ [4].

Варьируемые параметры модели:

1) $X = \left\{ x_{i,j,l}, i \in I, l \in [1:k_j],\ j \in J \right\}$, где $x_{i,j,l}$ – момент начала выполнения операции l работы j на станке i;

2) $Y = \left\{ y_{i,j,l} \in \{0,1\}, i \in I, l \in [1:k_j],\ j \in J \right\}$, где $y_{i,j,l} = 1$ если l-ая операция j-ой работы выполняется на i-ом станке; $y_{i,j,l} = 0$, если l-ая операция j-ой работы не выполняется на i-ом станке;

3) $Z = \left\{ z_{i,j,l} \in [0:N], i \in I, l \in [1:k_j],\ j \in J \right\}$, где $z_{i,j,l}$ - номер по порядку выполнения l-ой операции j-ой работы на i-ом станке.

В качестве ограничений модели рассмотрим следующие условия.

1) Каждая операция любой работы выполняется на одном станке

$$\sum_{i \in I} y_{i,j,l} = 1, l \in [1:k_j],\ j \in J.$$

2) Начало любой операции может наступить лишь после завершения всех операций, ей предшествующих по технологии:

ЕСЛИ $y_{i,j,l} = 1$ И $y_{s,j,l-1} = 1$, ТО $x_{i,j,l} \geq x_{s,j,l-1} + t_{s,j,l-1}, l \in [2:k_j], i,s \in I,\ j \in J.$

3) Начало выполнения любой работы на станке может начаться лишь после завершения выполнения на этом станке предыдущей работы:

$$\text{ЕСЛИ } y_{i,j,l} = 1 \text{ И } y_{i,v,s} = 1 \text{ И } z_{i,j,l} = z_{i,v,s} + 1,$$
$$\text{ТО } x_{i,j,l} \geq x_{i,v,s} + t_{i,j,l} + \tau_{i,v,j}, s \in [1:k_v], l \in [1:k_j], i \in I,\ j,v \in J.$$

4) Момент начала выполнения самой первой для станка операции может наступить лишь после наладки станка на эту работу

$$x_{i,j,l} \geq r_{i,j,l}, \text{ если } z_{i,j,l} = 1, i \in I, l \in [1:k_j],\ j \in J\ .$$

5) Естественные условия на введенные переменные

$$x_{i,j,l} \geq 0,\ y_{i,j,l} \in \{0,1\},\ z_{i,j,l} \in [0:N],\ i \in I,\ l \in [1:k_j],\ j \in J.$$

Очевидно, реализация указанных ограничений может быть осуществлена с использованием инструментальных средств систем управления проектами (СУП) [3].

2. Формулировка критериев оптимальности

В рамках методологии управления проектами в качестве основного критерия оптимальности рассматривается суммарная длительность проекта (в данном случае – время изготовления всей партии изделий) [3].

В то же время для производственных систем часто оказываются более важными другие показатели эффективности, что лежит в основе таких современных методологий управления и организации производства как «точно вовремя» (JIT, just-in-time) [2] и «бережливое производство» (lean production) [1]. К числу их важнейших концепций относится сокращение запасов и незавершённого производства, обеспечение непрерывности производственного процесса. Это требование можно формализовать как стремление привести сроки выполнения планируемых процессов (цепочек связанных задач) к их идеальной длительности с точки зрения технологии. Обозначим его как *суммарное время рассогласования (СВР)* $\sum \Delta t_i$ и потребуем его минимизации: $\sum \Delta t_i \to \min$. Время рассогласования для каждого процесса определяется как $\Delta t_i = t_{изгот.i} - t_{изгот.идеал.}$, где $t_{изгот.идеал.} = const$ – идеальное время выполнения технологического процесса, $t_{изгот.i}$ – планируемое время выполнения i-ого технологического процесса с учётом загрузки ресурсов (см. рис. 1) после их выравнивания с использованием стандартного инструментария СУП [3].

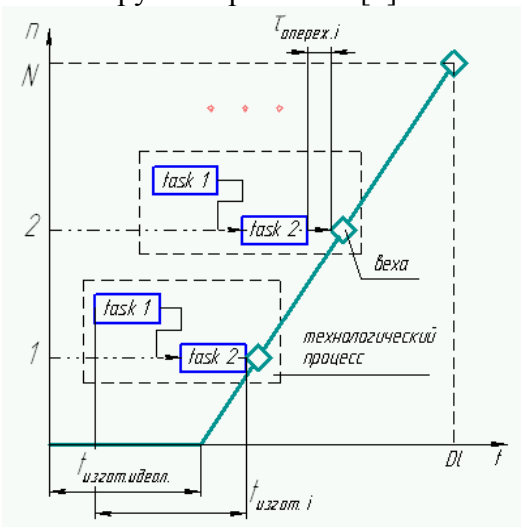

Рис. 1 - График производства партии изделий

Другой важной концепцией при организации производства является его *равномерность*. Это означает, что программа выпуска должна выполняться с одинаковым темпом, график выпуска при этом представляет собой прямую линию (см. рис. 1). Это требование можно формализовать как *суммарное время опережения (СВО)* $\sum \tau_i = \sum \tau_{опереж.i}$, $\sum \tau_i \to \min$; для фиксации идеального времени завершения очередного процесса используется механизм *вех* [3], при этом все задачи планируются с опцией *«начать как можно позже»*.

3. Жизненный цикл календарного планирования

Предлагаемый в работе вариант календарного планирования производства с учетом нестандартных критериев оптимальности состоит из 3 этапов, представленных ниже.

1 этап. Подготовка исходных данных. Представим, что есть технологический процесс, который состоит из связанных подзадач. Необходимо автоматически продублировать данный технологический процесс *N* раз (в соответствии с программой выпуска, которая должна быть выполнена к определенному сроку завершения проекта *Dl* (*deadline)*). После дублирования универсальные ресурсы (*generic resource*) (в производственных процессах это, как правило, станки, рабочие места) распределяются по подзадачам всех процессов.

2 этап. Начальная расстановка вех для реализации равномерности производства. Каждой последняя подзадача каждого процесса связывается с вехой, которая изначально фиксируется для случая идеально равномерного производства (см. рис. 1). Тем самым, изначально достигнут оптимум по критерию СВО.

3 этап. Применение алгоритма многокритериальной оптимизации. Выполняется оценка критерия СВР. Если оно не является приемлемым, осуществляется сдвиг вех, что увеличивает значение критерия СВО, но может сократить СВР. Вопрос о выбор метода оптимизации на текущий момент находится на стадии исследования.

4. Программная реализация

На данном этапе реализованы 1 и 2 этапы, рассмотренные выше. Были расширены возможности существующего пакета *Microsoft Project 2010* с точки зрения описания процессов, для чего реализовано приложение (add-in) на языке C#. Данное приложение позволяет дублировать множества однотипных проектов с автоматическим отслеживанием изменений событий их атрибутов (например, название задачи, ее длительность, связи с предшествующими и последующими задачами).

Для разработки приложения под *MS Project 2010* была выбрана среда *Visual Studio 2010*, язык программирования C# [5]. В *Visual Studio 2010* представлены шаблоны проектов, которые можно использовать для создания надстроек уровня приложения для *Microsoft Office Project*. Надстройки позволяют автоматизировать приложение *Project*, а также расширить функциональные возможности и настроить пользовательский интерфейс этого приложения.

При расширении приложения *Microsoft Office* путем создания надстройки уровня приложения код создается непосредственно для класса *ThisAddIn* данного проекта. Этот класс можно использовать для получения доступа к объектной модели ведущего приложения *Microsoft Office*, настройки пользовательского интерфейса приложения, а также для предоставления объектов созданной надстройки другим решениям *Office*.

Написание кода в проектах надстройки в некоторых аспектах отличается от написания кода в других проектах *Visual Studio*. Многие из этих отличий связаны с тем, каким образом объектные модели *Microsoft Office* представлены управляемому коду.

Создание кода надстройки начинается в классе *ThisAddIn*. Visual Studio автоматически создает этот класс в файле кода *ThisAddIn.cs* (для C#) проекта надстройки. Среда выполнения *Visual Studio Tools for Office* автоматически создает экземпляр этого класса при загрузке надстройки приложением *Microsoft Office*.

В классе *ThisAddIn* существует два обработчика событий по умолчанию. Для выполнения кода при загрузке надстройки следует добавить код в обработчик событий *ThisAddIn_Startup*. Для выполнения кода непосредственно перед выгрузкой надстройки следует добавить код в обработчик событий *ThisAddIn_Shutdown*.

Каждое приложение, которое может быть автоматизировано при помощи Visual Studio имеет объектную модель. В объектной модели приложения *Project* предоставляется доступ к различным типам, которые можно использовать для автоматизации *Project*. Эти типы позволяют создавать код для выполнения общих задач, таких как создание и изменение задач проекта программными средствами.

Для доступа к объектной модели Project из надстройки используется поле *Application* класса *ThisAddIn* проекта. В поле *Application* возвращается объект *Microsoft.Office.Interop.MsProject.Application* , который представляет текущий экземпляр *Project*.

5. Результат работы приложения

При запуске *MS Project 2010* всплывает дополнительное окно *Form1* (рисунок 2). С ячейкой для количества экземпляров и кнопкой *Duplicate*. В *Project* создаем свой проект, пусть он состоит, например, из трех задач, связанных между собой. Последняя задача – веха. Во вкладке «Лист ресурсов» назначаем задачам универсальные ресурсы *s1* и *s2*. Для лучшего понимания, пусть это будут станки 1 и 2, соответственно. Пусть *s1*=3, *s2*=2. Возвращаемся в Диаграмму Ганта и выделим при помощи мыши наши задачи, которые мы хотим объединить в одну группу и продублировать. Продублируем, например, наш проект 5 раз. Для этого вводим в ячейку число 5 и нажимаем на кнопку *Duplicate*. Получаем 5 идентичных проектов. Во время дублирования проекта происходит также дублирование ресурсов и вех.

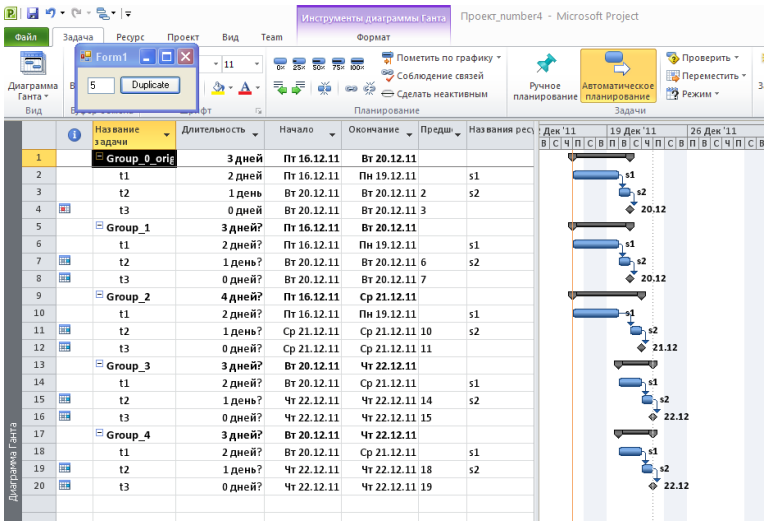

Рис. 2 - Результат работы приложения

Следует заметить распределение производственного процесса по универсальным ресурсам, число станков 1го типа было 3, а 2го – 2. Если требуется изменить Название *t1*, например на *t0*, то это название изменится во всех группах. Аналогично и с длительностью, началом и окончанием задач и подзадач.

Литература

1. «Your Online Resource for Lean-Based Information and Tools», Электронный ресурс, http://www.leanproduction.com/, доступ свободный.
2. William J. Stevenson, Системы «точно-в-срок» (just-in-time): разработка и внедрение, Электронный ресурс, http://www.elitarium.ru, свободный доступ.
3. Куперштейн В. «Microsoft Project 2010», изд. «БХВ-Петербург», Санкт-Петербург, 2011.
4. Прилуцкий М.Х., Власов С.Е. Многостадийные задачи теории расписаний с альтернативными вариантами выполнения работ. //Системы управления и информационные технологии, ООО "Издательство "Научная книга", 2005.
5. «Начните разрабатывать на платформе Майкрософт», Электронный ресурс, http://msdn.microsoft.com/ru-ru/, доступ свободный.