**ПАКЕТНОЕ ПЛАНИРОВАНИЕ В КЛИЕНТООРИЕНТИРОВАННОЙ ЦЕПИ ПОСТАВОК**

Esaignani Selvarajah George Steiner

McMaster University

(Получено 5 июля 2005 г .; пересмотрено 17 ноября 2005 г.)

**Аннотация** Планирование цепочки поставок - это новое направление исследований. Мы изучаем проблемы планирования поступления партий у производителя в многоуровневой клиентоориентированной цепочке поставок, где обещанные сроки выполнения работ считаются ограничениями, которые должны быть выполнены. Мы анализируем соотношение между затратами на хранение и затратами на доставку партии. Мы показываем, что проблемы тесно связаны с задачами пакетного планирования на одной машине с задачами, связанными с временем потока. Мы доказываем, что минимизация суммы общего взвешенного времени потока и затрат на доставку является строго NP-сложной. Для невзвешенной версии задачи мы представляем эффективные алгоритмы решения как для отдельных машин, так и для сборочных систем. Мы также разрабатываем динамическое программное решение для минимизации суммы максимального времени потока и затрат на доставку**.**

**Ключевые слова:** планирование, партии, цепочки поставок, ориентированные на клиента

1. **Введение**

За последние пятнадцать лет управление цепочками поставок было одной из важнейших тем исследований в сфере производства. Большая часть литературы по цепочке поставок фокусируется на вопросах управления запасами на стратегическом уровне с использованием стохастических моделей. Тем не менее, Томас и Гриффин [21] указывают в своем обзоре литературы, что для многих продуктов затраты на логистику могут составлять до 30% их стоимости. Это подчеркивает необходимость исследования задач цепочек поставок на оперативном уровне с использованием детерминированных, а не стохастических моделей. Недавно появившаяся область исследований в области планирования цепочек поставок пытается решить эту проблему.

Так как это новая область для исследований, существует относительно немного работ, конкретно посвященных проблемам планирования в цепочках поставок. Холл и Поттс [11] представили первую статью по этой важной теме. Они моделируют цепочку поставок, используя три уровня для потенциальных лиц, принимающих решения: поставщик, производитель(и) и клиенты. На рисунке 1 показана сеть для задачи с одним поставщиком (S), одним производителем (M) и m клиентами (C1, C2, ..., Cm). Производитель получает продукцию (задания) партиями, а время прибытия партии указано поставщиком. Производитель обрабатывает задания и доставляет готовую продукцию покупателям. Разделение заданий на партии и сроки поставки готовой продукции заказчикам определяется производителем для удовлетворения спроса на стадии планирования. Холл и Поттс [11] рассматривают каждый этап цепочки поставок как одну машину для целей планирования, и они изучают различные проблемы планирования, пакетирования и доставки с целью минимизации общего планирования и стоимости доставки. Чен и Холл [4] расширили модель до цепочек поставок с производственными системами сборочного типа. Selvarajah и Steiner [19, 20] детально изучили проблему планирования поставщика, когда целью является минимизация суммы запасов и затрат на доставку.

Общей чертой вышеупомянутых документов является то, что они предполагают, что тот, кто принимает решения о планировании, то есть поставщик или производитель, несет соответствующие расходы. Это означает, что после того, как решения по планированию были приняты на одном этапе, партии продуктов соответственно перемещаются через систему на следующий этап. Однако во многих современных производственных системах и цепочках поставок используется метод «тянуть», при котором решения о планировании принимаются на более поздней стадии, а продукты отбираются с предыдущей стадии, чтобы поступать в нужных количествах и точно в срок (JIT), когда они необходимы. JIT Manufacturing - это философия управления, которая стремится увеличить добавленную стоимость и устранить источники производственных отходов. Заинтересованный читатель в JIT отсылается к Monden [13] для более подробной информации. Еще одной ключевой особенностью систем JIT является то, что они стремятся сократить время выполнения заказа между этапами. Treville et. [23] утверждают, что координация цепочки поставок между партнерами без сокращения времени выполнения заказа может не улучшить работу цепочки до ее наилучшего возможного. Еще одним ключевым моментом успешного производства JIT является поддержание низкого уровня запасов при одновременном удовлетворении потребностей клиентов. Делается больший акцент на синхронизированном перемещении входов и выходов при производстве и доставке товаров и услуг клиентам. Частые поставки используются для сокращения запасов, что, конечно, увеличивает стоимость доставки. В этой статье мы изучаем этот компромисс в цепочках поставок, которые работают как вытягивающая система, т. Е. Получатель товара или деталей (покупатель на стадии 3 или производитель на стадии 2) принимает решения, как часто и когда партии продукты должны быть доставлены к нему. Мы предполагаем, что именно получатель всегда несет расходы, связанные с любым графиком доставки, который он или она решает. Это подразумевает, что будут избегаться неоправданно высокие частоты доставки при очень высоких затратах на доставку. Мы называем такие системы цепочками поставок, ориентированными на клиента, где для получателя используется общее слово «клиент».



Рисунок 1 - Сеть, показывающая отношения поставщика, производителя и клиента

Наше исследование мотивировано широким внедрением систем JIT во многих успешных производственных организациях. Одним из примеров является BMW, лауреат премии за производительность в области производства современных материалов [2]. Недавно расширенный завод BMW в Спартанбурге, штат Южная Каролина, использует тяговую систему для изготовления транспортных средств, указанных заказчиком, в течение 10 дней с момента размещения заказа. Хотя завод выпускает только две модели, внедорожники X-5 и двухместные родстеры Z-4, для каждой модели доступно множество вариантов с точки зрения формы, цвета и требований к интерьеру. Например, для модели X-5 есть 8 вариантов кузова, 12 цветов, 19 вариантов двигателя, 16 вариантов интерьера и 85 других вариантов. Завод постоянно информирует своих поставщиков о точных и стабильных данных о спросе. В результате завод может успешно следовать философии JIT.

В сегодняшней конкурентной среде наиболее важной задачей для цепочек поставок является своевременное удовлетворение спроса клиентов. Классическое планирование цепочки поставок принудительного типа пытается удовлетворить установленные сроки, назначая штрафы за опоздания или опоздания в расписание и пытаясь минимизировать эти штрафы. В цепочке поставок, ориентированной на клиента, существует ограничение на соблюдение всех сроков. Фактически, эти сроки исполнения (окончательные графики поставки) предполагаются как входные данные, и они управляют системой.

В нескольких работах в литературе были изучены задачи составления расписаний для одной машины с ограничениями по срокам. Смит [22] разрабатывает алгоритм с полиномиальным временем для суммы задачи времени завершения с ограничениями по срокам, а Хек и Робертс [9] представляют алгоритм для минимизации суммы времен завершения при условии, что максимальная задержка не увеличивается. Ленстра и соавт. Показали, что взвешенная сумма проблемы времени завершения была сильно NP-сложной. [12]. Для этой проблемы в литературе появился ряд алгоритмов ветвления и привязки. К ним относятся алгоритмы Поттса и Ван Вассенхове [16], Познера [15] и Вернера [25]. Относительно наиболее эффективным является недавний алгоритм Пана [14].

В модели, которую мы изучаем, есть N заданий, J1, J2, ..., JN, которые должны быть обработаны на заводе-изготовителе, чья система может быть смоделирована либо одной машиной, либо операцией типа сборки с подзадачами Ji, j, которые должны быть обработано на l станках в серии для i = 1, ..., N и j = 1, ..., l. Работа Ji должна быть доставлена ​​клиенту вовремя Di. Стоимость этих поставок несет заказчик. В модели с одной машиной Ji требует обработки в течение времени pi для i = 1, 2, ..., N. В операции сборки время обработки подзадачи Ji, j обозначается через pi, j. (Если задание пропускает определенную операцию, то pi, j = 0 для соответствующей подзадачи.) Поскольку задание не доставлено до его крайнего срока, производитель хочет выполнить его как можно ближе к этим крайним срокам. Поэтому мы предполагаем, что задания обрабатываются в порядке ранней даты исполнения (EDD), и это приводит к выполнимому графику, т. е. У производителя есть достаточные возможностей, чтобы сделать этот график целесообразным для соблюдения сроков. Производитель получает детали и расходные материалы для каждой работы или подзадачи от своего поставщика(ов) партиями и оплачивает стоимость доставки d для каждой партии. Производитель должен вовремя получать партии, чтобы он мог уложиться в конечные сроки, но он не хочет получать расходные материалы слишком рано, потому что каждая работа Ji связана с затратами на хранение в интервале времени [ai, Di], где ai - его время прибытия у производителя при i = 1, 2, ..., N. Стоимость владения запасами работы Ji тесно связана с ее временем потока, определенным как Di - ai. Поскольку стоимость доставки измеряется в денежном выражении, мы умножаем показатели производительности, связанные со временем потока, на соответствующие константы, чтобы обеспечить совместимость измерений. Поэтому мы умножаем сумму времен потока на постоянную h, которая является стоимостью удержания работы в инвентаре за единицу времени; умножить максимальное время потока на постоянную K, которая является штрафом, связанным с максимальным временем потока; и умножаем время потока задания Ji на wi, которое является стоимостью удержания задания Ji в единицу времени, когда целью является минимизация суммы взвешенных времен потока и затрат на доставку. Без ограничения общности можно предположить, что последовательность заданий J1, J2, ..., JN. Затем производитель хочет найти оптимальное время поступления aj для каждого задания Jj, количество партий n и разбиение заданий на партии поступления, что определяет размеры пакетов, чтобы общая стоимость была минимизирована. Мы предполагаем, что у производителя всегда есть выполнимый график, т. е.

Мы рассматриваем следующие цели:

1. Для суммы времени потока с пакетированием, общая стоимость
2. Для максимального времени потока с пакетированием, общая стоимость
3. Для суммы взвешенных времен потока с пакетированием, общая стоимость

Проблемы пакетного планирования широко изучались ранее. Поттс и ван Вассенхове [18], Альберс и Брукер [1], Вебстер и Бейкер [24], Поттс и Ковалев [17] дали исчерпывающие обзоры. Несмотря на то, что было много документов, только некоторые из них связаны с проблемами пакетного планирования с расходами на доставку. Cheng et al. [5] Изучили планирование пакетов на одном компьютере, чтобы минимизировать сумму затрат на доставку и штрафов за досрочное использование. Ян [26] анализирует аналогичную модель с заданными датами поставки партии и Hall et al. [10] изучили связанные проблемы в различных машинных средах. Чен [3] представляет алгоритм динамического программирования для планирования на одной машине и общего назначения сроков со штрафами за опоздание и затраты на доставку партии.

Статья продолжается следующим образом. В следующем разделе мы изучим проблемы планирования поступления партий, чтобы минимизировать общее взвешенное время потока и затраты на доставку, то есть функцию стоимости TC3. Сначала мы докажем, что проблема сильно NP-сложная на одной машине даже с общим сроком выполнения для всех заданий. После этого мы представляем алгоритм динамического программирования с линейным временем для задачи о фиксированной последовательности поступления задания. Этот алгоритм неоднократно используется в Разделе 3 для минимизации TC1 как для одиночной машины, так и для среды сборочного цеха. В разделе 4 мы представляем эффективный алгоритм динамического программирования для планирования поступления партий с целью TC2. Последний раздел содержит наши заключительные замечания.

1. **Планирование пакетного прибытия для минимизации общего веса времени потока и затрат на доставку**
   1. **Сложность**

Давайте рассмотрим проблему планирования поступления партий у производителя, когда его система моделируется одной машиной, и цель состоит в том, чтобы минимизировать TC3.

**Теорема 2.1** Минимизация сильно NP-сложная.

**Доказательство:** Холл и Поттс [11] доказали, что минимизация суммы суммарных взвешенных времен потока и затрат на доставку для поставщика в системе push-типа сильно NP-сложная. Они использовали хорошо известную проблему NP-hard 3-PARTITION, чтобы свести ее к своей задаче планирования. Мы показываем, как это сокращение можно адаптировать, чтобы доказать сильную NP-сложность нашей проблемы.

3-PARTITION [8]:

Даны 3r целых числа u1,…,u3, где существует ли раздел A1,…,Ar из набора индексов {1, ..., 3r}, такой что | Aj | = 3 и , для j = 1,…,r?

Рассмотрим следующий случай нашей задачи с расписанием: N = 3r, задание Ji имеет pi = wi = ui и Di = rz, для i = 1, ..., 3r, d = z2 / 2 и пусть C = r (r + 2) z2 / 2 будет пороговым значением. Мы доказываем, что для этого экземпляра существует расписание прибытия партии тогда и только тогда, когда существует решение для 3-PARTITION.

Предположим, что 3-PARTITION имеет решение, и предположим без ограничения общности, что целые числа пронумерованы так, что u3i − 2 + u3i − 1 + u3i = z, для i = 1,..., r. Рассмотрим расписание, в котором задания планируются в этой последовательности J1, ..., JN и поставляют пакет Bi для заданий {J3i − 2, J3i − 1, J3i}, поступающих в то время, когда J3i − 2 начинает свою обработку, то есть a3i −2 = a3i − 1 = a3i = (i − 1) z, для i = 1,..., r. Легко увидеть, что каждое из заданий {J3i − 2, J3i − 1, J3i} имеет время потока, равное rz− (i − 1) z = (r − i + 1) z. Следовательно, для этого графика.

Далее мы докажем теорему в другом направлении. Предположим, у нас есть расписание с n партиями поступления, и пусть xi будет общим временем обработки заданий, соответствующих i-му пакету Bi для i = 1, ..., n. Ясно, что поставки для Bi должны прибыть в то время, когда последняя работа в Bi-1 завершает свою обработку, т.е. . Следовательно, время выполнения работ в Bi будет i=1,…,n. Таким образом мы имеем . Таким образом, минимизация TC3 для n партий может быть записана как минимизация

при условии, что

Поскольку первое слагаемое этой цели - (rz)2, легко увидеть, что вся функция будет минимизирована, когда x1 = x2 = ... = xn = rz / n. Таким образом, для любого n-пакетного решения мы должны иметь TC3 ≥ (rz)2/2 + n (rz/n)2/2 + nz2/2. Простые аргументы из исчисления показывают, что это выражение достигает своего минимума при n = r, а минимальное значение равно C. Таким образом, если существует расписание пакетирования с TC3 = C, то мы должны иметь n = r и каждый пакет должен иметь размер xj. = z. Это означает, что в каждом пакете есть 3 задания, а у 3-PARTITION есть решение.

* 1. **Пакетирование заданной последовательности заданий на одном компьютере**

В этом разделе мы изучаем проблему оптимального пакетирования у производителя, чтобы свести к минимуму , когда задан порядок обработки задания и прибытия к производителю, и это также порядок поступления задания. Без ограничения общности пусть эта последовательность будет J1, ... JN. Обратите внимание, что данная последовательность обработки задания предполагается выполнимой для удовлетворения обещанных сроков доставки. Пусть Si обозначает самое позднее время начала работы Ji, так что график выполним. Рассмотрим n-пакетное расписание прибытия, и пусть ij будет индексом первого задания для пакета Bj прибытия, то есть, пакетное расписание: {i1, i1 + 1, ..., i2 - 1}, {i2, i2 + 1, ..., i3 - 1}, ..., {in, in + 1, ..., N}. Тогда легко увидеть, что партия Bj должна прибыть во время , а не раньше. Таким образом

Обратите внимание, что мы предполагаем, что для этой задачи существует выполнимый график, поэтому выполняется уравнение 1.1.

Если мы определим SN+1 = DN, то мы можем написать Следовательно

Пусть Sj+1 – S = p’j для j = 1,2,…,N. Обратите внимание, что pj ≥ pj и pj можно интерпретировать как отрезок времени на машине, «выделенной» для задания j. (У нас есть pj > pj, если в расписании есть время простоя.) Тогда

Расширяя результаты Coffman et al. [7], Альберс и Брукер [1] представили алгоритм O(N) кратчайшего времени для поиска оптимального пакетирования, чтобы минимизировать сумму взвешенных времен завершения фиксированной последовательности заданий на одном компьютере. Мы показываем, что минимизация TC3 также может быть сформулирована как особая проблема кратчайшего пути путем обмена времен обработки для весов и весов для выделенных времен pj в формулировке Альберса-Брукера:

Поскольку первая сумма является константой, достаточно минимизировать только вторую сумму. Если мы установим

Тогда Таким образом, cij представляет вклад в стоимость пакета, содержащего задания Ji, Ji + 1, ..., Jj − 1. Кроме того, для любого k> j ≥ i,

Поскольку f(i) монотонно не возрастает и h (j,k) > 0 для всех j < k, наша задача также эквивалентна специальной задаче кратчайшего пути, обсужденной Альберсом и Брукером. Ниже приводится соответствующая сеть:

а. Каждое задание Ji (i = 1, 2, ..., N) представлено вершиной i в сети.

b. Сеть содержит направленные ребра (i, j) для всех пар i < j.

c. Ребру (i, j) назначается длина ребра cij.

d. Добавлена фиктивная работа JN + 1 с pN + 1 = wN + 1 = 0.

Легко увидеть, что минимизация TC3 эквивалентна нахождению кратчайшего пути от вершины 1 до вершины N + 1 в сети.

Пусть Fj - длина кратчайшего пути от вершины j до вершины N + 1, а Fj(k) - длина кратчайшего пути от j до N + 1, который содержит (j, k) в качестве первого ребра. Затем,

Fj(k) = cjk + Fk + d, и Fj = min{Fj(k)|j < k ≤ N + 1}.

Кроме того, Fj(k) ≤ Fj(l) для вершин j < k < l эквивалентно Fj(k) - Fj(l) = cjk + Fk - cjl - Fl ≤ 0. Поскольку cjl - cjk = f(j) h(k,l), получаем, что



Рисунок 2 – Структура очереди q

Таким образом, для любых двух вершин k < l, если порог , тогда Fj(k) ≤ Fj(l) и l называется не лучше, чем k относительно j. С другой стороны, если f(j) < δ (k, l), то Fj(k) > Fj(l) и l называется лучше, чем k относительно j. Основываясь на этих свойствах монотонности, мы представляем Алгоритм 1, который является слегка измененной версией алгоритма Альберса и Брукера, адаптированной к нашей задаче. Он использует структуру данных очереди, показанную на рисунке 2.

**Алгоритм 1** Модифицированный алгоритм Альберса и Брукера

begin

Step 1 q = N + 1; FN+1 = 0

Step 2: for j = N to 1 do begin

Step 3: while head(q) = tail(q) and f(j) ≥ δ(next(head(q)), head(q)) do

Delete head(q) from q

Step 4: N(j) = head(q); Fj = cjN(j) + FN(j)

Step 5: while head(q) = tail(q) and δ(j, tail(q)) ≤ δ(tail(q), previous(tail(q))) do

Delete tail(q) from q

Step 6: Add j to the tail of q

end

end

Корректность алгоритма и его временная сложность O(N) могут быть доказаны так же, как в [1].

**Теорема 2.2** Алгоритм 1 вычисляет за O(N) время оптимальную дозировку, которая минимизирует на заданной последовательности задания.

Отметим, что алгоритм 1 также применим для минимизации целевой функции TC1.

1. **Минимизация общего времени потока и затрат на доставку**

В этом разделе мы изучаем одновременное упорядочение и пакетирование заданий для прибытия к производителю, чтобы свести к минимуму .

* 1. **Планирование прибытия пакета на одну машину**

**Лемма 3.1** Существует оптимальный график для всех трех целей, TC1, TC2, TC3, в котором порядок прибытия на работу такой же, как и порядок обработки работы.

**Доказательство:** задание не может начать обработку до тех пор, пока задание, непосредственно предшествующее ему в последовательности обработки EDD, не будет завершено, и прибытие любого задания до прибытия задания, предшествующего ему в последовательности обработки, может только увеличить общее время потока. Следовательно, никакая работа не должна поступать в оптимальном графике раньше, чем кто-либо из ее предшественников в порядке EDD.

Без потери общности мы индексируем задания в том порядке, в котором они находятся в этой общей последовательности поступления и обработки. Тогда самое позднее возможное время начала задания Ji, Si, может быть рекурсивно вычислено по SN = DN - pN, а Si = min {Di, Si + 1} - pi для i = N - 1, N - 2, ... , 2, 1.

Хотя последовательность поступления и обработки задания одинакова, оптимальный график может содержать задания, которые приходят рано и ожидают в магазине. Рассмотрим следующий пример минимизации TC1 при h = 1:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| J | 1 | 2 | 3 | 4 | 5 | 6 |
| pj | 9 | 7 | 5 | 12 | 6 | 5 |
| Dj | 23 | 23 | 23 | 48 | 48 | 48 |

Допустим, нам дана последовательность обработки задания {1,2,3,4,5,6}, и мы должны найти оптимальный график 2-пакетирования. Для этой задачи оптимальное дозирование составляет B1 = {1, 2, 3, 4} и B2 = {5, 6} с общим временем потока 131. В этом решении для дозирования вторая партия приходит в момент времени t = S5 = 37, а обработка набора заданий {5, 6} завершается в момент времени t = 48; первая партия приходит в момент времени t = S1 = 2, и обработка набора заданий {1, 2, 3} завершается в момент времени t = 23, а обработка набора заданий {4} завершается в момент времени t = 35. Если мы переместим задание J4 ко второй партии, затем B2 должен прийти к t = 25, так что набор заданий {4, 5, 6} будет завершен к t = 48; и набор заданий {1, 2, 3} должен прибыть в t = 2 и будет завершен в момент времени t = 23. Общее время выполнения этого нового расписания увеличится до 132. Это показывает, что в оптимальном расписании некоторые задания могут прибыть с ранними партиями и ждать в магазине.

**Лемма 3.2** Существует оптимальный график для всех трех целей TC1, TC2 и TC3, в котором партия поступает только тогда, когда все ранее доступные задания у производителя завершили обработку, т. е. Партия поступает только тогда, когда машина доступна для начала его обработки.

**Доказательство:** поскольку задана последовательность обработки заданий, задание не может быть запущено до завершения непосредственно предшествующего задания. Из леммы 3.1 порядок поступления заданий следует последовательности обработки. Поэтому задания, назначенные для любого пакета, будут обработаны после того, как последнее задание предыдущего пакета завершило обработку. Таким образом, прибытие партии до завершения обработки последней работы предыдущей партии прибытия может только увеличить время выполнения расписания.

**Лемма 3.3.** Пусть задание Jj будет первым заданием, которое будет обработано в партии поступления Bk, тогда партия Bk должна прибыть в момент Sj для всех трех целей TC1, TC2 и TC3.

**Доказательство.** Предположим, что существует оптимальный график, который не удовлетворяет лемме, но соответствует леммам 3.1 и 3.2. Выберите последнюю партию, которая прибывает до последнего времени начала первой работы партии в этом расписании. Если мы перенесем время прибытия партии на самое позднее время начала первой работы этой партии, график останется выполнимым. Кроме того, время выполнения всех заданий, принадлежащих этой партии, будет сокращено, а время выполнения всех заданий, принадлежащих другим партиям, останется неизменным. Это противоречит предположению оптимальности для графика.

**Лемма 3.4** Существует оптимальный график для всех трех целей, TC1, TC2 и TC3, в котором задания, которые должны быть доставлены заказчику (-ам) в один и тот же срок, запланированы в LPT (сначала самое большое время обработки) заказа у производителя.

**Доказательство:** мы знаем, что задания, которые должны быть доставлены заказчикам в одно и то же время, обрабатываются последовательно у производителя из-за заказа на обработку EDD. Не ограничивая общности, пусть эта последовательность будет J1, ..., JN. Предположим, что лемма неверна для оптимального расписания. Тогда будет не менее двух заданий Ji и Ji + 1 у производителя, так что pi <pi + 1 и Di = Di + 1. Если задания Ji и Ji + 1 принадлежат одной и той же партии поступления, то их замена не повлияет на время выполнения какой-либо работы. Если задания Ji и Ji + 1 принадлежат разным партиям поступления, скажем, Bk и Bk + 1 соответственно, то пусть время прибытия партии Bk будет равно tk, а время прибытия Bk + 1 будет равно tk + 1. Из леммы 3.3 мы знаем tk + 1 = min {Di + 1, Si + 2} −pi + 1. Поменяйте местами задания Ji и Ji + 1 в этих пакетах, что не изменит размеры пакета поступления. Назовите новые партии Bk и Bk + 1. Таким образом, в новом расписании Bk достигает tk, а Bk + 1 достигает tk + 1 = min {Di, Si + 2} - pi = min {Di + 1, Si + 2} - pi = tk + 1 + pi + 1 − pi Обратите внимание, что обмен не повлияет на выполнимость графика. Обмен не повлияет на время выполнения заданий в Bk \ Ji в исходном расписании. Время выполнения каждго задания в Bk + 1 \ Ji + 1 уменьшается на pi + 1 - pi > 0 по сравнению с исходным расписанием. Время потока Ji + 1 увеличивается на tk + 1 - tk, а время потока Ji уменьшается на tk + 1 - tk > tk + 1 - tk. Таким образом, чистое изменение общего времени потока - это уменьшение по меньшей мере на pi + 1 – pi, что противоречит оптимальности исходного графика. Следовательно, любые задания Ji и Ji + 1, находящиеся не в порядке LPT, должны принадлежать одному и тому же пакету. Повторное повторение последовательности заданий с одинаковой датой исполнения в порядке LPT внутри пакетов не меняет стоимость или выполнимость графика и приводит к оптимальному графику, удовлетворяющему условиям леммы.

Комбинация упорядочения EDD с упорядочением заданий LPT с одинаковой датой исполнения внутри пакетов устанавливает оптимальную последовательность для заданий. Поскольку мы знаем последовательность заданий, алгоритм 1 можно использовать для поиска оптимальных размеров партии.

**Алгоритм 2:** Алгоритм минимизации суммы времени потока и затрат на доставку

Шаг 1: Заказать работы в порядке EDD и запланировать работы с той же датой исполнения в порядке LPT.

Шаг 2: Вызовите Алгоритм 1, чтобы найти оптимальные размеры пакета поступления найденной последовательности.

**Теорема 3.1** Алгоритм 2 находит за время O(NlogN) оптимальный график поступления партии, который минимизирует TC1, сумму времени потока и затрат на доставку.

**Доказательство:** шаг 1 находит оптимальную последовательность заданий путем сортировки, которая требует O(NlogN) времени. Алгоритм 1 находит оптимальное пакетирование этой последовательности обработки задания за O(N) времени.

* 1. **Планирование прибытия партии для сборочного цеха**

В этом разделе мы изучим оптимальную политику поступления пакетов в сборочный цех, где задания обрабатываются и собираются на серии l машин. На каждой машине для работы могут потребоваться детали, поставленные одним из q поставщиков. Схема цепочки поставок для этой проблемы показана на рисунке 3. Производитель должен поставлять нужную продукцию в нужных количествах в обещанные сроки для клиентов, и расходы по доставке взимаются с клиентов. Чтобы соответствовать обещанным срокам поставки, производитель должен заказать детали у поставщиков, обработать и собрать их на станках. Для любого продукта Ji (i = 1, 2, ..., N), который не требует обработки на j-й машине (j = 1, 2, ..., l), мы устанавливаем pi,j равным нулю. Поставщики должны доставлять детали производителю в требуемое время, а расходы на доставку от поставщиков деталей производителю отправляются на счет производителя. Таким образом, производитель хочет найти оптимальные графики поступления партий для деталей от каждого поставщика, чтобы минимизировать общую сумму потоков и затрат на доставку при соблюдении обещанных сроков доставки клиентам.



Рисунок 3: Сеть, показывающая отношения производителя с q поставщиками и m клиентами

Пусть Si,j обозначает самое позднее возможное время начала задания Ji,j в допустимом графике. Тогда тот факт, что задания «протягивают» свои подзадачи через систему, можно уловить с помощью следующих обратных рекурсивных вычислений:

**Лемма 3.5** Существует оптимальный график поступления партий и связанный график производства, в котором задача Ji, j (для i = 1, 2, ..., N; j = 1, 2, ..., l) начинает свою обработку во время Si, j.

**Доказательство.** Используя Si + 1,l в качестве верхней границы времени завершения задачи Ji,l, первые две строки (3.1) гарантируют, что на последней машине будет достаточно времени для завершения обработки Ji, l по Di. Расчеты в последнем ряду (3.1) обеспечивают достаточное время и для задачи Ji, j на машине j при i = 1, 2, ..., N; j = 1, 2, ..., l. (График выполним, если Si,1 ≥ 0 для i = 1, 2, ..., N.) Также ясно, что время прибытия деталей для Ji, j, ai,j, должно удовлетворять ai,j ≤ Si, j, чтобы график был осуществим, но некоторые части могут прибыть рано. Теперь предположим, что у нас есть оптимальный график, в котором есть некоторые задачи, которые запускаются до их последнего времени запуска. Рассмотрим последнюю такую задачу, скажем Jr, k, и переместим ее обработку, чтобы начать с Sr, k. Смена не повлияет ни на выполнимость графика, ни на время выполнения какой-либо задачи. Повторное применение приведенного выше аргумента ко всем оставшимся ранним задачам приведет к графику, который удовлетворяет лемме.

**Лемма 3.6** Задачи планирования поступления партий от каждого поставщика отделимы (т. е. График прибытия от поставщика не зависит от графиков прибытия от других поставщиков) и может решаться независимо друг от друга.

**Доказательство.** Из леммы 3.5 мы знаем, что существует оптимальный график производства, при котором каждая задача Ji, j запускается в самое позднее возможное время начала Si,j. Тогда Si,j можно рассматривать как крайний срок для доставки деталей, необходимых от их поставщика. Поскольку каждая задача Ji,j получает свою часть(и) от не более одного поставщика по предположению, каждый Si,j может стать крайним сроком поставки только для одного поставщика. Таким образом, независимо от того, какое время прибытия партии запланировано от поставщика, это не влияет на время потока других деталей (задач) от других поставщиков. Таким образом, рассматривая требования к доставке от одного поставщика, мы получаем отдельную задачу планирования прибытия партий для этого поставщика. Поэтому задачи могут быть решены отдельно друг от друга для каждого поставщика.

**Теорема 3.2** Задача планирования поступления партий у производителя сборки может быть оптимально решена за время O(qlN log(lN)).

**Доказательство.** Согласно лемме 3.6, детали, поступающие от каждого поставщика, могут быть запланированы для поставки в виде отдельной задачи планирования партии. У нас есть (не более) q из этих проблем. Каждый из них может быть решен с помощью алгоритма 2 за O(lN log(lN)), таким образом, общее требуемое время действительно не более O(qlN log(lN)).

1. **Минимизация максимального времени потока и затрат на доставку**

В этом разделе мы рассмотрим задачу планирования поступления партий с целью производителя. Легко видеть, что леммы 3.1-3.3 применимы и к этой проблеме, и их можно доказать аналогичным образом.

**Лемма 4.1.** Существует оптимальный график, при котором задания, которые должны быть доставлены заказчикам в тот же срок, запланированы в заказе LPT у производителя.

**Доказательство:** пусть будет оптимальный график, при котором задания с одинаковой датой исполнения не следуют порядку LPT. Без ограничения общности пусть последовательность заданий будет J1, J2, ..., JN. Тогда будет как минимум два задания Ji и Ji+1 с pi < pi + 1 и Di = Di + 1. Если Ji и Ji + 1 принадлежат одному и тому же пакету, то чередование этих двух заданий не повлияет на максимальное время потока. Если задания Ji и Ji + 1 принадлежат разным партиям поступления, скажем, Bk и Bk + 1 соответственно, то пусть время прибытия партии Bk будет равно tk, а время прибытия Bk + 1 будет равно tk + 1. Из леммы 3.3 мы знаем, что tk + 1 = min {Di + 1, Si + 2} - pi + 1. Поменяйте местами задания Ji и Ji + 1 в этих пакетах, не меняя размеры пакетов поступления. Назовите новые партии Bk и Bk + 1. Таким образом, в новом расписании Bk достигает tk, а Bk + 1 достигает tk + 1 = min {Di, Si + 2} - pi = min {Di + 1, Si + 2} - pi = tk + 1 + pi + 1 – pi. Обратите внимание, что обмен не повлияет на выполнимость графика. Кроме того, обмен не повлияет на время выполнения заданий в Bk \ Ji в исходном расписании. Время выполнения каждой работы в Bk + 1 \ Ji + 1 уменьшается на pi + 1 - pi > 0 по сравнению с исходным расписанием, а время выполнения Ji явно уменьшается. Единственное увеличенное время потока - это время Ji + 1, которое увеличивается на tk + 1 - tk < Di + 1 - tk. Мы имеем, однако, Di + 1 - tk = Di - tk, и последний является временем потока Ji в исходном расписании. Следовательно, максимальное время потока нового расписания не будет больше, чем у исходного. Повторение этого обмена для каждого нарушения порядка работы в лемме даст оптимальный график, удовлетворяющий его условиям.

Обратите внимание, что лемма подразумевает, что существует последовательность заданий, которая является оптимальной как для максимального времени потока плюс стоимость доставки, так и для суммы времени потока плюс цели стоимости доставки. Чтобы найти оптимальное прибытие дозирования для тем не менее, мы не можем использовать алгоритм 1, который был разработан для суммы целей времени потока. Поэтому ниже мы представляем новый алгоритм динамического программирования.

**Алгоритм 3.** Алгоритм минимизации в последовательности заданий J1, J2,…, JN.

Пусть f (k, j) будет минимальным значением TC2 для первых j заданий в расписании, использующих k партий поступления для 1 ≤ k ≤ j ≤ N. Для более простых обозначений мы также определяем f (k, j) = ∞ для 1 ≤ j <k ≤ N. Оптимальное значение TC2 можно получить с помощью mink = 1, ..., N f (k, N). Рекурсивное вычисление f (k, j) для 1 ≤ k ≤ j ≤ N определяется следующим образом.

где Si (k, j-1) - время начала первого задания, i(k, j-1), последней партии в расписании, реализующей f(k, j-1). Первая строка рекурсии соответствует выбору оптимального расписания, реализующего f(k - 1, r), и добавлению к нему новой партии поступления, содержащей задания {r + 1, ..., j} для r = k - 1,. .., j - 1. Здесь [f (k - 1, r) - (k - 1) d] / K выражает максимальное время потока расписания, реализующего f (k - 1, r). Второй ряд рекурсии соответствует случаю, когда задание Jj просто добавляется в последний пакет расписания, реализуя f(k, j - 1) без запуска нового пакета. Чтобы облегчить вычисления, нам нужно сохранить индекс первого задания последнего пакета в расписании, реализующем f(k, j), обозначаемом i(k, j).

Начальные условия: f (0, 0) = 0 и f (k, j) = ∞ для j, k = 1, ..., N.

**Теорема 4.1** Алгоритм 3 находит оптимальный график поступления партий у производителя, чтобы минимизировать общую стоимость максимального времени потока и времени доставки за O(N3).

**Доказательство.** Алгоритм должен вычислять значения O(N2) f(k,j). Каждое вычисление требует времени O(N). Сохраняя индексы i(k,j), мы можем получить оптимальное пакетирование в конце путем возврата.

1. **Обобщение и заключительные замечания**

Мы изучили проблемы с планированием поступления партий у производителя в ориентированной на клиента цепочке поставок, где обещанные сроки выполнения работ считаются ограничениями, которые должны быть выполнены. Мы показали, что проблемы тесно связаны с проблемами пакетного планирования на одной машине с задачами, связанными с временем потока. Мы доказали, что минимизация суммы общего взвешенного времени потока и затрат на доставку сильно NP-сложная. Для невзвешенной версии задачи мы представили эффективные алгоритмы решения как для отдельных машин, так и для сборочных систем. Мы также разработали динамическое программное решение для минимизации суммы максимального времени потока и затрат на доставку.

Дальнейшие исследования в этой области могут искать альтернативные целевые функции или искать эффективные эвристические или приближенные решения для вычислительно сложного взвешенного случая.

**Благодарность**

Это исследование было частично поддержано Канадским советом по естественным и техническим исследованиям в рамках гранта Discovery 1798-03.

**Библиографический список**

[1] S. Albers and P. Brucker: The complexity of one-machine batching problems. Discrete

Applied Mathematics, 47 (1993), 87-107.

[2] BMW: Custom cars on demand - The automaker uses a pull system to build customer-

specified vehicles within 10 days of order placement. Modern Materials Handling, (2004).

[3] Z.L. Chen: Scheduling with batch setup times and earliness-tardiness penalties. Euro-

pean Journal of Operational Research , 96 (1997), 518-537.

[4] Z.L. Chen and N.G. Hall: Supply chain scheduling - Assembly systems. Working Paper,

(2005).

[5] T.C.E. Cheng, V.S. Gordon and M.Y. Kovalyov: Single machine scheduling with batch

deliveries. European Journal of Operational Research, 94 (1996), 277-283.

[6] S. Chopra, and P. Meindl: Supply Chain Management: Strategy, Planning, and Opera-

tion ( Prentice Hall, Toronto, 2004).

[7] E.G. Coffman, M. Yannakakis, M.J. Magazine, and C. Santos: Batch sizing and job

sequencing on a single machine. Annals of Operations Research, 26 (1990), 135-147.

[8] M.R. Garey and D.S. Johnson: Computers and Intractability: A Guide to the Theory

of NP-Completeness (W.H. Freeman, San Francisco, 1979).

[9] H. Heck and S. Roberts: A note on the extension of a result on scheduling with secondary

criteria. Naval Research Logistics Quarterly, 19 (1972), 403-405.

[10] N.G.Hall, M.A. Lesaoana and C.N.Potts: Scheduling with fixed delivery dates. Opera-

tions Research, 49 (2001), 134-144.

[11] N.G. Hall and C.N. Potts: Supply chain scheduling: Batching and delivery. Operations

Research, 51 (2003), 566-584.

[12] Lenstra, J. K. A. H.G. Rinnooy Kan and P. Brucker: Complexity of machine scheduling

problems. Annals of Discrete Mathematics, 1 (1977), 342-362.

[13] Y. Monden: Toyota production system. (Second Edition, 1993).

[14] Y. Pan: An efficient exact algorithm for single machine scheduling with due dates

to minimize total weighted completion time. Operations Research Letters, 31 (2003),

492-496.

[15] M.E. Posner: Minimizing weighted completion times with deadlines. Operations Re-

search, 33 (1985), 562-574.

[16] C.N. Potts, and L. N. Van Wassenhove: An algorithm for single machine sequenc-

ing with deadlines to minimize total weighted completion time. European Journal of

Operational Research, 12 (1983), 379-387.

[17] C.N. Potts and M.Y. Kovalyov: Scheduling with batching - A review. European Journal

of Operational Research, 120 (2000), 228-249.

[18] C.N Potts and L.N.Van Wassenhove: Integrating scheduling with batching and lot-

sizing - a review of algorithms and complexity. Journal of Operational Research Society,

43 (1992), 395-406.

[19] E. Selvarajah and G. Steiner: Batch scheduling in a two-level supply chain - A focus

on the supplier. European Journal of Operational Research, to appear (2004).

[20] E. Selvarajah and G. Steiner: Batch Scheduling at the Upstream Supplier to Minimize

Delivery and Inventory Holding Costs, submitted, (2005).

[21] D.J. Thomas and P.M. Griffin: Coordinated supply chain management. European Jour-

nal of Operational Research, 94 (1996), 1-15.

[22] W.E. Smith: Various optimizers for single-stage production. Naval Research Logistics,

(1956) Quarterly 3, 59-66.

[23] S. Treville, R. D. Shapiro and A. Hameri: From supply chain to demand chain - the role

of lead time reduction in improving demand chain performance. Journal of Operations

Management, 21 (2004), 613-627.

[24] S. Webster, and K.R. Baker: Scheduling groups of jobs on a single machine. Operations

Research, 43 (1995), 692-703.

[25] F. Werner: A branch and bound algorithm for minimizing weighted completion times

with deadlines. Optimization, 28 (1993), 187-199.

[26] X. Yang: Scheduling with generalized batch delivery dates and earliness penalties. IIE

Transactions, 32 (2000), 735-741.