|  |  |
| --- | --- |
|  |  |
| **Математическая модель  процессов гомогенизации и резки**   |  |  | | --- | --- | | **Разработано:** | **Институт Автоматизации проектирования РАН** | | **Версия документа:** | **0.1** | | **Дата создания:** | **09.01.2014** | | |
| **Москва – 2013** | |

**Контроль изменений документа**

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя изменившего** | **Дата изменения** | **Версия** | **Описание изменения** |
| Нелюбин А.П. | 09.01.2014 | 0.1 | Первоначальная версия документа.  В предположении возможности складывать заготовки на пол. |

**оглавление**

1. Математическая модель процессов гомогенизации и резки 4

1.1. Общее описание модели 4

1.2. формальное описание модели 5

2. Алгоритм моделирования процессов 6

2.1. Общее описание алгоритма 6

2.2. формальное описание алгоритма 7

1. **Математическая модель процессов гомогенизации и резки**
   1. **Общее описание модели**

Цель моделирования процессов гомогенизации и резки – выявление рациональных схем загрузки линий Хертвич.

Основная задача: по заданной схеме загрузки определить число заготовок, которые приходится складывать на пол из-за перегрузки линии.

Также модель нужна для вычисления времени окончания обработки заготовок.

Процессы гомогенизации и резки моделируются после процессов литья, и мы уже знаем информацию о поступающих партиях заготовок (когда, сколько заготовок, какие).

**Принципиальное устройство линии Хертвич:**

1. **Первый (входной) буфер** с точки зрения моделирования является плоской платформой (без ячеек), на которой помещается определенное число заготовок в зависимости от их диаметра.
2. **Печь гомогенизации** содержит постоянно движущийся конвейер с 158 ячейками под заготовки. Скорость движения конвейера зависит от диаметра заготовок: чем толще, тем медленнее.
3. **Второй (промежуточный) буфер** состоит из нескольких отделов: выходной накопитель печи, камера охлаждения, входной накопитель пилы. Охлаждение не является узким местом процесса, поэтому второй буфер с точки зрения моделирования – такая же плоская платформа, на которой помещается определенное число заготовок в зависимости от диаметра.
4. **Резка** заготовок на слитки производится по одной. Число резов зависит от заказа (сколько слитков в заготовке). Скорость реза зависит от диаметра заготовки: чем толще, тем медленнее.

**Описание моделируемого процесса:**

Сразу после окончания литья считаем, что заготовки мгновенно поступают на первый или второй (если не нужна гомогенизация) буфер. Если часть заготовок не помещается в первый буфер, то они складываются на пол. Количество сложенных на пол заготовок суммируется для учета штрафа.

Каждое поступление заготовок вносится в компьютер. Компьютер знает, какие есть заготовки, сколько их, где (на какой стадии) они находятся. В соответствии с этой информацией компьютер контролирует загрузку заготовок в печь гомогенизации:

1. Толстые заготовки помещаются на конвейер через одну ячейку.
2. При переходе с тонких заготовок на толстые обеспечивается необходимый интервал в 14 ячеек.
3. При переходе с толстых заготовок на тонкие загрузка останавливается, пока не будут выгружены все толстые слитки из печи.
4. Компьютер наперёд просчитывает переполнение второго буфера и заранее останавливает загрузку заготовок в печь, чтобы этого переполнения не произошло. Дело в том, что заготовки нельзя передерживать в печи.

Если в конвейере печи находятся заготовки разного диаметра, то скорость движения определяется по последней загруженной заготовке.

На ЛО №1 САЗ на два литейных агрегата имеется две линии Хертвич. Считаем, что заготовки с каждого из этих агрегатов можно отправлять на любую из этих линий. Кроме того, одну ходку можно поделить на 2 части и отправить на разные линии Хертвич. Это внутренняя задача оптимизации загрузки линий Хертвич, которую можно решать отдельно.

Заготовки забираются с пола по остаточному принципу. В первую очередь обрабатываются вновь поступающие заготовки, чтобы избежать лишних перекладываний. Когда забирать заготовки с пола – также можно решать путем внутренней оптимизации. Ясно, что толстые заготовки с пола не следует пускать между тонкими, и наоборот.

Однако в первой версии алгоритма предлагается не нарушать порядок следования заготовок, то есть пускать заготовки с пола сразу за своей партией.

* 1. **ФОРМАЛЬНОЕ описание модели**

**Объект**: линия гомогенизации и резки *hc*

**Известные параметры** линии гомогенизации и резки:

- <list>*CU*(*hc*) – список литейных агрегатов (может быть 2 ЛА)

- <map>*Tcell*(*hc*, *diameter*) – время перемещения ячейки конвейера на одну позицию (сек)

- <map>*Tcut*(*hc*, *diameter*) – время одного реза (сек)

- *nCells*(*hc*) – число ячеек на конвейере печи

- <map>*nBlanksB*1(*hc*, *diameter*) – номинальное число заготовок, помещающееся в первый буфер

- <map>*nBlanksB*2(*hc*, *diameter*) – номинальное число заготовок, помещающееся во второй буфер

- *R*(*hc*) – общее число ремонтов литейной машины в месяц

- <list>*TSr*(*hc*) и *TFr*(*hc*) – время начала и окончания ремонта *r* = 1, …, *R*(*hc*)

**Объект**: новая партия заготовок *P* (ходка или часть ходки)

**Известные параметры** новой партии:

- *Diameter*(*P*) – диаметр заготовок (мм)

- *nBlanks*(*P*) – количество заготовок в партии (~ 60-120)

- *nIngots*(*P*) – количество слитков в одной заготовке

- *Cast*(*P*)) – соответствующая ходка

- *HC*(*P*) – линия Хертвич, на которую поступает партия заготовок

- *T*(*P*) = (*Cast*(*P*)) – время поступления новой партии на линию Хертвич (время окончания литья соответствующей ходки)

- *Homogenization*(*P*) – нужна ли гомогенизация (если нет, то заготовки отправляются сразу во второй буфер)

**Объект**: заготовка *blank*

**Известные параметры** заготовки:

- *HC*(*blank*) – линия Хертвич, на которую поступает заготовка

- *P*(*blank*) – партия (ходка) заготовки

- *Diameter*(*blank*) = *Diameter*(*P*(*blank*)) – диаметр заготовки (мм)

- *nIngots*(*blank*) = *nIngots*(*P*(*blank*)) – количество слитков в заготовке

**Вычисляемые параметры** заготовки:

- (*blank*) – момент времени загрузки заготовки в печь.

- (*blank*) – момент времени выгрузки заготовки из печи.

- (*blank*) – момент загрузки заготовки в печь.

- (*blank*) – момент загрузки заготовки в печь.

- *Next*(*blank*) – указатель на следующую заготовку на линии Хертвич

- *CellsAfter*(*blank*) – количество пропусков ячеек после заготовки

1. **алгоритм моделирования процессов**
   1. **общее описание алгоритма**

Для решения основной задачи необходимо знать расположение заготовок на линии Хертвич в момент поступления каждой новой партии заготовок. Из-за принятого уровня детализации процессов необходимо моделировать движение каждой заготовки по линии Хертвич. В обоих буферах важно знать только последовательность расположения заготовок. Внутри печи гомогенизации, помимо этого, нужно знать, в каких ячейках находятся заготовки.

Предлагается следующий подход к моделированию процесса. По данным о поступающих партиях заготовок на линию Хертвич *hc* выстраиваем единую последовательность заготовок *bj* с номерами *j* = 1, …, *J*. Определяем количество пропусков ячеек после каждой заготовки, учитывая, что второй буфер не должен переполниться. Эта информация однозначно задает движение заготовок по линии Хертвич. Тогда мы сможем посчитать, сколько заготовок находится в первом буфере при поступлении каждой новой партии, а также вычислить время окончания обработки каждой заготовки.

Сложность представляет моделирование переполнения второго буфера. Если моделировать процесс движения заготовок по линии Хертвич последовательно, то мы не знаем заранее, когда переполнится второй буфер, и когда останавливать загрузку заготовок в печь. Предположим, что возникло переполнение второго буфера, так что мы не можем поместить на него заготовку *blank*, выходящую из печи. В этом случае заготовку *blank* следует передвинуть назад так, чтобы она вышла из печи как раз в тот момент, когда второй буфер сможет ее принять. Предыдущие заготовки также нужно передвинуть назад, причем часть из них вернется из печи обратно в первый буфер. В результате это может повлиять на переполнения первого буфера. Задача состоит в том, чтобы определить число дополнительных ячеек, возникающих между заготовками в результате таких возвратов.

Также следует учесть партии заготовок, поступающие сразу во второй буфер, без гомогенизации. Если в момент их поступления происходит переполнение второго буфера, то есть два варианта. Первый – считать такое расписание недопустимым. Второй – передвигать гомогенизируемые заготовки из второго буфера обратно в печь, считая число дополнительных пропусков ячеек.

Получается, что итоговая последовательность заготовок с пропусками ячеек между ними определяется в несколько этапов. Сначала учитываются только гомогенизируемые заготовки: определяются пропуски между ними при загрузке в печь, определяются времена окончания процессов гомогенизации и резки для каждой заготовки. Затем учитывается ограничение на длину второго буфера, определяются дополнительные пропуски, пересчитывается времена. Затем в последовательность встраиваются партии негомогенизируемых заготовок. Затем считаются переполнения первого буфера.

* 1. **ФОРМАЛЬНОЕ описание алгоритма**

**Шаг 1.** Выстраиваем первоначальную последовательность заготовок *bj* на основе информации о поступающих партиях *Pi*. Проставляем для каждой заготовки число пропусков ячеек вслед за ней *CellsAfter*(*bj*): между толстыми – 1, между тонкими – 0, при переходе от толстой к тонкой – 158, при переходе от тонкой к толстой – 14. Это начальные значения, которые потом будут меняться.

**Шаг 2.** Считаем последовательно для всех заготовок, начиная с первой, время загрузки в печь гомогенизации:

, где

Если между партиями образуется долгий простой, такой что

, где *bj* – первая заготовка в партии *Pi*,

то считаем число дополнительных пропусков после последней заготовки из предыдущей партии:

При этом полезно сохранить и старые значения *CellsAfter*(*bj*-1), так как они могут понадобиться, когда мы будем двигать заготовки назад при учете второго буфера.

**Шаг 3.** Представим второй буфер в виде окна, скользящего от начала последовательности заготовок. Сначала в голове окна находится заготовка *b*1. Используя информацию *nBlanksB*2(*hc*, *diameter*), находим число *L* последующих заготовок, помещающихся в это окно.

**Шаг 4.** Считаем последовательно для всех заготовок в окне, начиная с первой, время выгрузки из печи гомогенизации. Если подошло время партии негомогенизируемых заготовок, то вклиниваем ее здесь в последовательность заготовок, пересчитываем *L*.

**Шаг 5.** Считаем последовательно для всех заготовок в окне, начиная с первой, время начала и окончания резки:

,

,

,

.

**Шаг 6.** Положим *j* = 1. *L* – текущее число заготовок в окне.

**Шаг 7.** Пытаемся удалить из окна заготовку *bj*, смотрим, помещается ли при этом в окно заготовка *bL*+*j*. Если нет, то пытаемся удалить из окна следующую заготовку: положим *j* = *j*+1, *L* = *L* – 1, повторяем Шаг 7. Если да, то переходим на Шаг 8.

**Шаг 8.** Считаем время выгрузки из печи заготовки *bL*+*j*. Если подошло время негомогенизируемой партии, то переходим на Шаг 9, иначе на Шаг 10.

**Шаг 9.** Пытаемся вклинить новую партию в последовательность перед заготовкой *bL*+*j*. Если эта партия полностью помещается во второй буфер, то просто переходим на Шаг 7. Если нет, то в варианте 1 считаем такое расписание вообще недопустимым; в варианте 2 пытаемся вклинить новую партию перед заготовкой *bL*+*j-*1 и т.д., пока не добьемся того, что новая партия поместится во второй буфер. Затем нужно пересчитать длину окна *L* и перейти на Шаг 7.

**Шаг 10.** Если выполняется условие ,

то происходит переполнение второго буфера. Считаем число ячеек, на которые следует подвинуть назад заготовку *bL*+*j*, а также последующие заготовки. Добавляем их к *CellsAfter*(*bL*+*j*-1).

**Шаг 11.** Помещаем в окно заготовку *bL*+*j*. При необходимости пересчитываем для нее время загрузки и выгрузки из печи, а также считаем время начала и окончания резки.

**Шаг 12.** Также следует проверить, помещается ли в окно заготовка *bL*+*j*+1 (такое возможно, когда *bj* толще *bL*+*j* и *bL*+*j* +1). Если это так, то помещаем ее в окно, считаем времена выгрузки из печи, начала и окончания резки, а затем увеличиваем *L* = *L* + 1.

**Шаг 13.** Пока не кончились заготовки *L*+*j* ≤ *J*, перемещаем окно *j* = *j*+1 и переходим на Шаг 7. Иначе переходим на Шаг 14.

**Шаг 14.** Уточняем время загрузки в печь всех заготовок, считаем переполнения первого буфера.