**Вопрос 1**

**Параллельное программирование** означает достаточно широкую область, которая связана с организацией расчетов на вычислительных системах, состоящих из нескольких процессорных устройств.

**Необходимость параллельного программирования:**

* Приложения требуют увеличения производительности компьютеров.
* Производительность процессора и памяти ограничена физическими характеристиками применяемых материалов.
* Задачи часто содержат независимые компоненты, которые могут решаться одновременно (т. е. параллельно).

**Параллельные вычисления применяются в областях, связанных с проведением больших расчетов:**

* Системах поддержки проектирования (CAD – Computer Aided Design).
* Инженерных приложениях.
* Математическом моделировании физических процессов.
* Моделирование глобальных процессов в науке о Земле.
* Вычислительной химии.
* Бизнес-приложениях.

**Конвейеризация –** способ повышения производительности процессора, основанный на разделении подлежащей исполнению команды на более мелкие части, называемыми этапами или ступенями, и выделении для каждой из них отдельного блока аппаратуры.

**Векторизация –** вид распараллеливания программы, при котором однопоточные приложения, выполняющие одну операцию в каждый момент времени, модифицируются для выполнения нескольких однотипных операций одновременно.

**Параллелизм –** это свойство систем, при котором несколько вычислений выполняются одновременно, и при этом, возможно, взаимодействуют друг с другом. Вычисления могут выполняться на нескольких ядрах одного чипа с вытесняющим разделением времени потоков на одном процессоре, либо выполняться на физически отдельных процессорах.

**Примеры параллельных вычислительных систем:**

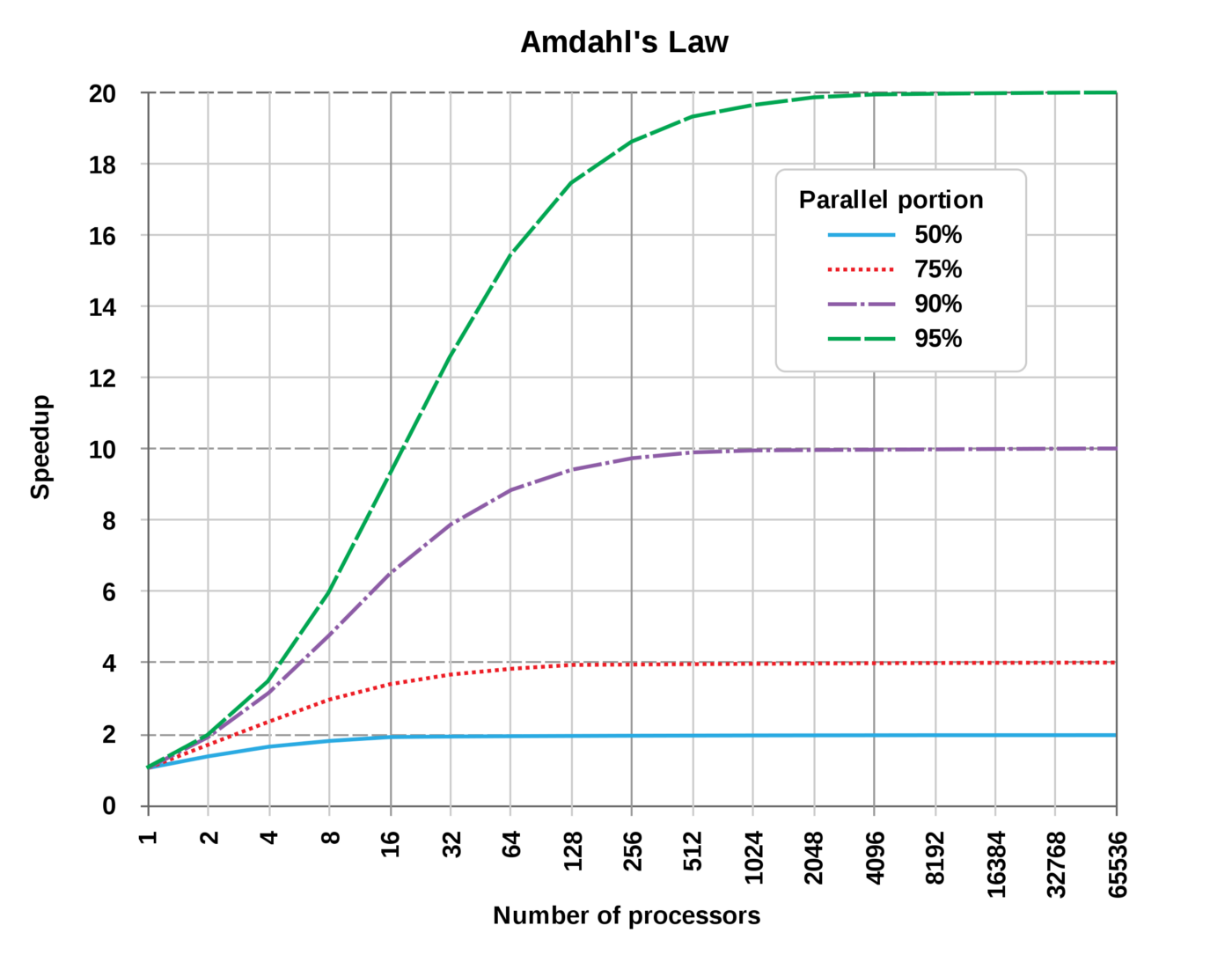
* **Суперкомпьютер –** специализированная вычислительная машина, значительно превосходящая по своим техническим параметрам и скорости вычислений большинство существующих в мире компьютеров. Как правило, современные суперкомпьютеры представляют собой большое число высокопроизводительных серверных компьютеров, соединённых друг с другом локальной высокоскоростной магистралью для достижения максимальной производительности в рамках подхода распараллеливания вычислительной задачи.
* **Кластер –** группа компьютеров, объединённых высокоскоростными каналами связи, представляющая с точки зрения пользователя единый аппаратный ресурс. **Кластер –** слабо связанная совокупность нескольких вычислительных систем, работающих совместно для выполнения общих приложений, и представляющихся пользователю единой системой.

Обычно различают следующие **основные виды кластеров**:

* отказоустойчивые кластеры (High-availability clusters, HA, кластеры высокой доступности).
* кластеры с балансировкой нагрузки (Load balancing clusters).
* вычислительные кластеры (High performance computing clusters, HPC).
* системы распределенных вычислений.

**Вопрос 5**

**Закон Амдала –** иллюстрирует ограничение роста [производительности](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%BC%D0%BE%D1%89%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%B0) [вычислительной системы](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B) с увеличением количества [вычислителей](https://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BD%D1%82%D1%80%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80). [Джин Амдал](https://ru.wikipedia.org/wiki/%D0%90%D0%BC%D0%B4%D0%B0%D0%BB,_%D0%94%D0%B6%D0%B8%D0%BD) сформулировал закон в 1967 году, обнаружив простое по существу, но непреодолимое по содержанию ограничение на рост производительности при [распараллеливании вычислений](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F): «В случае, когда задача разделяется на несколько частей, суммарное время её выполнения на [параллельной системе](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B) не может быть меньше времени выполнения самого медленного фрагмента». Согласно этому закону, ускорение выполнения программы за счёт [распараллеливания](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D0%BF%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D0%B8%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC) её [инструкций](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%86%D0%B8%D1%8F_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) на множестве вычислителей ограничено временем, необходимым для выполнения её последовательных инструкций.



Пусть необходимо решить некоторую вычислительную задачу. Предположим, что её [алгоритм](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC) таков, что доля *{\displaystyle \alpha }αα* от общего объёма вычислений может быть получена только последовательными расчётами, а, соответственно, доля 1 – *α* {\displaystyle 1-\alpha }1-1111 может быть распараллелена идеально (то есть время вычисления будет обратно пропорционально числу задействованных узлов p{\displaystyle p}pp). Тогда ускорение, которое может быть получено на вычислительной системе из p {\displaystyle p} процессоров, по сравнению с однопроцессорным решением не будет превышать величины

**Закон Густавсона-Барсиса** **–** оценка максимально достижимого ускорения выполнения параллельной программы, в зависимости от количества одновременно выполняемых потоков вычислений («процессоров») и доли последовательных расчётов.

Закон Густавсона-Барсиса выражается формулой

*s* – доля последовательных расчётов в программе,

*n* – количество процессоров.

Данную оценку ускорения называют ускорением масштабирования, так как данная характеристика показывает, насколько эффективно могут быть организованы параллельные вычисления при увеличении сложности решаемых задач.

**Отличие от закона Амдала**

При оценке ускорения параллельного выполнения закон Амдала предполагает, что объём задачи остаётся постоянным. Величина ускорения по закону Амдала показывает, во сколько раз меньше времени потребуется параллельной программе для выполнения. Однако величину ускорения можно рассматривать и как увеличение объёма выполненной задачи за постоянный промежуток времени. Закон Густавсона-Барсиса появился именно их этого предположения.

**Анализ масштабируемости параллельных вычислений**

Целью применения параллельных вычислений во многих случаях является не только уменьшение времени выполнения расчетов, но и обеспечение возможности решения более сложных вариантов решаемых задач (таких постановок, решение которых не представляется возможным при использовании однопроцессорных вычислительных систем). Способность параллельного алгоритма эффективно использовать процессоры при повышении сложности вычислений является важной характеристикой выполняемых расчетов. В связи с этим, параллельный алгоритм называют масштабируемым (scalable), если при росте числа процессоров он обеспечивает увеличение ускорения при сохранении постоянного уровня эффективности использования процессоров. Возможный способ характеристики свойств масштабируемости состоит в следующем. Оценим накладные расходы (total overhead), которые имеют место при выполнении параллельного алгоритма

Накладные расходы появляются за счет необходимости организации взаимодействия процессоров, выполнения некоторых дополнительных действий, синхронизации параллельных вычислений и т. п. Используя введенное обозначение, можно получить новые выражения для времени параллельного решения задачи и соответствующего ускорения:

Используя полученные соотношения, эффективность использования процессоров можно выразить как

Последнее выражение показывает, что, если сложность решаемой задачи является фиксированной (*T1* = const), то при росте числа процессоров эффективность, как правило, будет убывать за счет роста накладных расходов *T0*. При фиксации числа процессоров эффективность использования процессоров можно улучшить путем повышения сложности решаемой задачи *T1* (предполагается, что при росте параметра сложности *n* накладные расходы *T0* увеличиваются медленнее, чем объем вычислений *T1*). Как результат, при увеличении числа процессоров в большинстве случаев можно обеспечить определенный уровень эффективности при помощи соответствующего повышения сложности решаемых задач. В этой связи, важной характеристикой параллельных вычислений становится соотношение необходимых темпов роста сложности расчетов и числа используемых процессоров.

Пусть *E*=const есть желаемый уровень эффективности выполняемых вычислений. Из выражения для эффективности можно получить

или

Порождаемую последним соотношением зависимость *n=F(p)* между сложностью решаемой задачи и числом процессоров обычно называют функцией изоэффективности (isoefficiency function).

Покажем в качестве иллюстрации вывод функции изоэффективности для учебного примера суммирования числовых значений. В этом случае

и функция изоэффективности принимает вид

Как результат, например, при числе процессоров p=16 для обеспечения уровня эффективности *E*=0.5 (т.е. *K*=1) количество суммируемых значений должно быть не менее *n*=64. Или же, при увеличении числа процессоров с *p* до *q* (*q>p*) для обеспечения пропорционального роста ускорения необходимо увеличить число суммируемых значений *n* в раз.

{\displaystyle S\_{p}={\cfrac {1}{\alpha +{\cfrac {1-\alpha }{p}}}}}