Комуникация от точка до точка

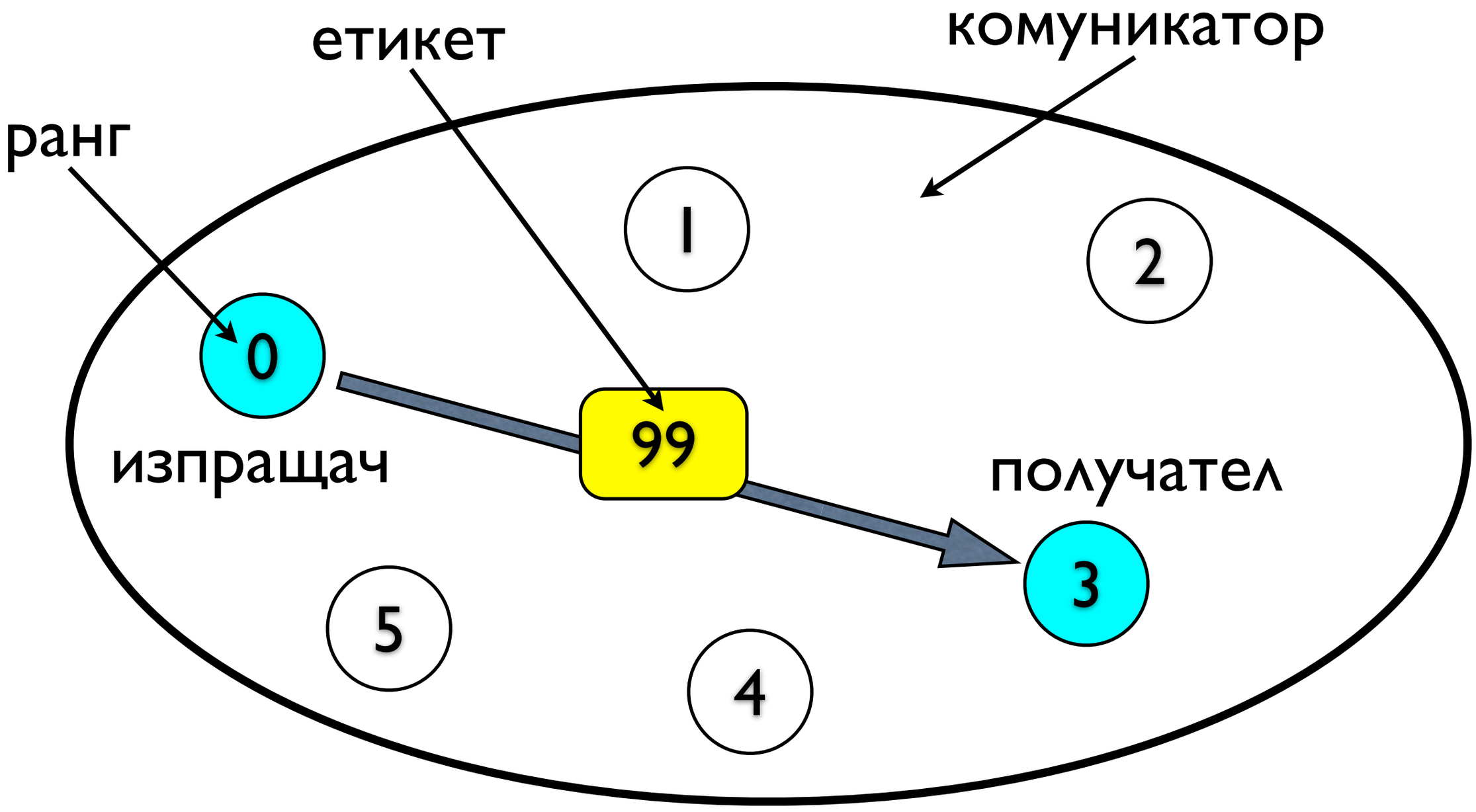
Увод в паралелното програмиране #4

Материалите са разработени в рамките на договор ДО-02-136/2008 с Фонд Научни Изследвания
Проект: IRC-CoSiM Ръководител на проекта: проф. дфзн Ана Пройкова, ФзФ на СУ
http://www.irc-cosim.uni-sofia.bg/

Общи положения

- Базов механизъм за обмяна на информация между процесите в паралелната задача
- Процес *източник* предава на процес *приемник*
- Всеки от двата процеса извиква съответната MPI операция
- Комуникацията се осъществява в рамките на определен *комуникатор*, в който членуват и двата процеса

Комуникация в картинки



Видове комуникатори

- Комуникация в рамките на една група от процеси – *интракомуникатор*
- Комуникация между различни групи от процеси – *интеркомуникатор*
- *Локална* и *отдалечена* група
- Интеркомуникацията позволява логическо разделение на дейностите в рамките на една паралелна задача

Режими на комуникация

- 4 различни комуникационни режима:
  - синхронен
  - буфериран
  - стандартен
  - режим с готовност
- Режимите се отнасят до операцията по изпращане
- Приемането винаги е синхронно
- Операциите приключват тогава, когато буферът за данни е свободен за използване от програмата

Режими на комуникация

	Условие за приключване
Синхронно изпращане	Приключва само след приключване на операцията по приемане
Буферирано изпращане	Винаги приключва (ако не възникне грешка), независимо дали приемането е приключило
Стандартно изпращане	Държи се като синхронно или буферирано според конкретната MPI реализация
Изпращане с готовност	Винаги приключва (ако не възникне грешка), независимо дали приемането е успешно
Приемане	Приключва след получаване на съобщението

Форми на извикванията

- Блокираща форма
  - MPI извикването блокира и не връща управлението на програмата до приключване на операцията
- Неблокираща (асинхронна) форма
  - MPI извикването връща управлението веднага и продължава да се изпълнява успоредно с изпълнението на програмата
  - връщат манипулатор на асинхронната операция, с който се тества или изчаква приключването

Форми и режими

Операция	Блокираща форма	Неблокираща форма
Синхронно изпращане	MPI_Ssend	MPI_Issend
Буферирано изпращане	MPI_Bsend	MPI_Ibsend
Стандартно изпращане	MPI_Send	MPI_Isend
Изпращане с готовност	MPI_Rsend	MPI_Irsend
Приемане	MPI_Recv	MPI_Irecv

Синхронно изпращане

MPI_Ssend (buf, count, dtype, dest, tag, comm)

- Приключва след започване на операцията по приемане от страна на приемащия процес
- Най-бавната и сигурна операция по изпращане
- Синхронизиращ ефект
- Гарантира се, че няма предишни съобщения в транзит
- Улеснява откриването на грешни допускания в комуникационните схеми

Буферирано изпращане

MPI_Bsend (buf, count, dtype, dest, tag, comm)

- Приключва след като съобщението бъде копирано в предварително прикачен потребителски буфер
- Буферът трябва да се задели от потребителя и да се прикачи с *MPI_Buffer_attach(buffer, size)*
- След приключване на употребата буферът трябва да се разкачи с *MPI_Buffer_dettach(buffer, size)*
- Предсказуемо поведение при претоварване на комуникационната система – грешка от препълване на буфера

Стандартно изпращане

MPI_Send (buf, count, dtype, dest, tag, comm)

- Приключва след започване на процеса на изпращане
- На практика се реализира като синхронно или буферирано изпращане спред реализацията и типа на съобщението
- Изпратените и неполучени съобщения се натрупват в опашки и буфери, препълването на които може да причини проблеми
- Не трябва да се разчита на преносимост на конкретното поведение на стандартното изпращане между различните реализации

Приемане

MPI_Recv (buf, count, dtype, src, tag, comm, status)

- Приключва след получаване на съобщението
- Приемащият буфер трябва да може да побере поне *count* елемента от тип *dtype*
- Приемат се само съобщения от *src* с маркер *tag* по реда на изпращането им
- Не се прави проверка за съвпадение на типа на данните *dtype*
- Приемат се съобщения, изпратени във всеки един от 4-те режима

Приемане

- Приемане на съобщения от произволен източник
  - *dst* == *MPI_ANY_SOURCE*
- Приемане на съобщения с произволен маркер
  - *tag* == *MPI_ANY_TAG*
- *status* съдържа конкретните стойности на:
  - източник: *status.MPI_SOURCE* (C)
    status(MPI_SOURCE) (Fortran)
  - маркер: *status.MPI_TAG* (C)
    status(MPI_TAG) (Fortran)
- *MPI_STATUS_IGNORE*, ако статусът не ни интересува

Проверка за съобщения

MPI_Probe (src, tag, comm, status)

- Приключва след наличие на съобщение със съвпадащи параметри
- Позволява преглед на съобщенията преди същинското им приемане
- Следващият *MPI_Recv* или *MPI_Probe* ще получи същото съобщение при съвпадение на *src* и *tag* с тези в *status*

Запитване за съобщението

MPI_Get_count(status, dtype, count)

- Връща дължината на съобщението в елементи от даден тип
- *count* == *MPI_UNDEFINED*, ако броят на елементите не е цяло число
- Проверката се прави с допускане, че изпращащият процес е указал тип *dtype* – самите съобщения не носят информация за типа на данните в тях
- Може да се използва след *MPI_Recv* или *MPI_Probe*

Изпращане с готовност

MPI_Rsend (buf, count, dtype, dest, tag, comm)

- Приключва след започване на процеса на изпращане
- Успява само, ако получателят на съобщението вече е извикал *MPI_Recv* или *MPI_Irecv*, преди започване на изпращането
- Поведението в останалите случаи не е дефинирано
- В някои случаи премахва част от комуникационния свръхтовар, но изисква специфична програмна логика и предварителна синхронизация

Неблокиращи операции

MPI_Isend (buf, count, dtype, dest, tag, comm, req)
MPI_Irecv (buf, count, dtype, src, tag, comm, req)

- Връщат веднага управлението и продължават да се изпълняват асинхронно
- В *req* се записва манипулатор на асинхронната операция
- *req* може да се тества за завършеност или да се изчака синхронно до приключването на операцията
- Буферът не трябва да се променя до приключване на асинхронната операция!

Изчакване и проверка

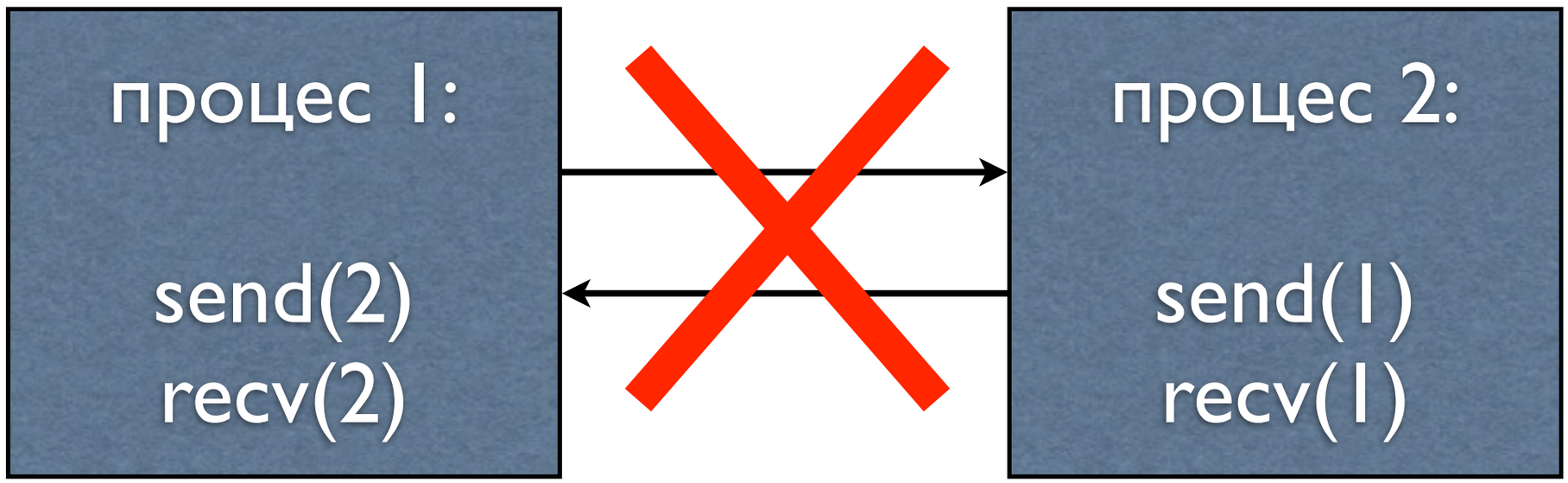
- *MPI_Test(req, flag, status)*
  - тества операция *req* и връща булев флаг за завършеност *flag*
  - *status* съдържа статуса на операцията в случай на завършеност (само за операции по приемане)
- *MPI_Wait(req, status)*
  - изчаква до приключване на операцията *req*
  - *status* съдържа статуса на операцията (само за операции по приемане)
- Версии *all*, *any* и *some* за множество манипулатори

Семантика на комуникациите

- Изпратените от един до друг процес съобщения се приемат в реда на изпращането им, независимо от използвания режим на изпращане
- При наличие на съвпадащи операции по изпращане и приемане, поне една от двете ще приключи нормално и няма да “зависне”
- Блокиращите и неблокиращите операции могат да се смесват свободно и от двете страни

Дизайн на комуникацията

- Не трябва да се допуска, че стандартното изпращане приключва преди да е започнало приемането – не винаги *MPI_Send* е буфериран



взаимно блокиране (deadlock)

- Възможността за блокиране се избягва с използване на *MPI_Sendrecv*

Дизайн на комуникацията

- Не трябва да се допуска, че стандартното изпращане приключва след като е започнало приемането – не винаги *MPI_Send* е синхронен
- Не трябва да се разчита, че изпратено преди това съобщение е пристигнало някъде, за коректната интерпретация на следващи съобщения
- Възможен е недетерминизъм във времево отношение в реда на получаване на съобщенията при участие на повече от два процеса

Дизайн на комуникацията

- Всички изпратени съобщения трябва да се “изконсумират” в съответстващи операции на приемане, иначе комуникационната система се “задръства”
- Добре написаните програми продължават да работят след замяна на *MPI_Send* с *MPI_Ssend*
- Допускането, че *MPI_Send* работи еднакво във всички MPI реализации е грешно и води до непреносимост на програмите