

Message Passing Interface

F.CS306 Parallel programming – Lecture 14

Topic Overview

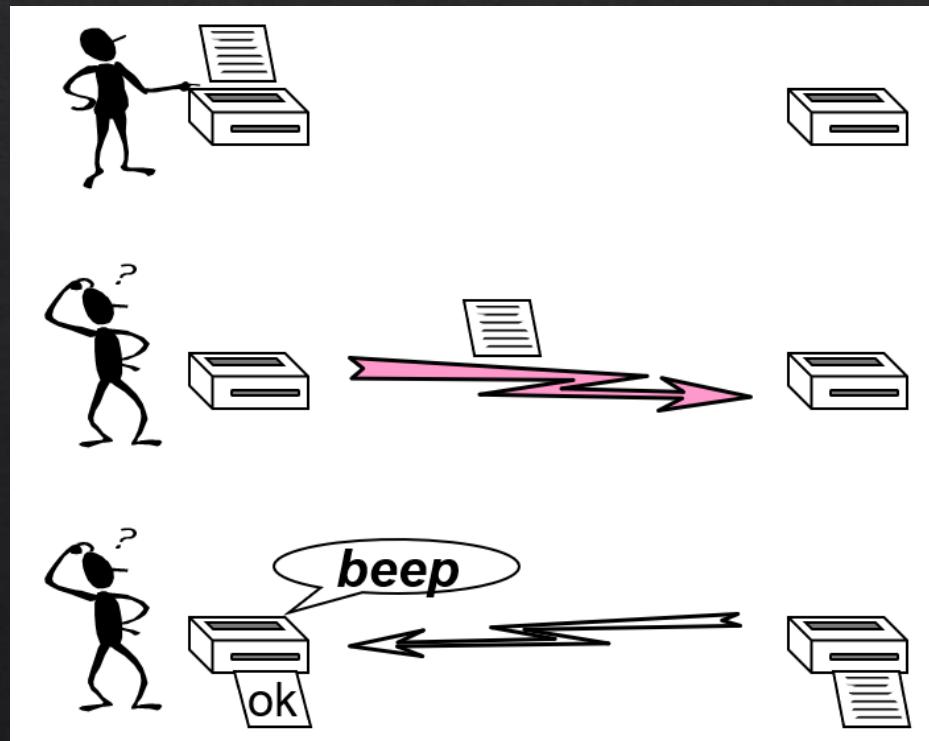
- ❖ Blocking болон Non-Blocking үйлдэл
- ❖ Мессеж ба Point-to-Point мэдээлэл солилцоо
 - ❖ Мессеж
 - ❖ Point-to-Point харилцаа
 - ❖ P-to-P мэдээллэ солилцооны горим
 - ❖ Мессеж дараалал хадгалалт
- ❖ Non-blocking мэдээлэл солилцоо
 - ❖ Deadlock
 - ❖ Non-Blocking send/receive

Blocking Operations

- ❖ Илгээгч/хүлээн авагч нь өөр процесийн үйлдэл хүртэл хүлээнэ:
 - ❖ synchronous send үйлдэл нь хүлээн авсан байх хүртэл хүлээдэг.
 - ❖ хүлээн авах үйлдэл нь мессеж илгээх хүртэл хүлээдэг.
- ❖ Blocking цикл зөвхөн тухайн үйлдэл гүйцэтгэгдсэний дараа л буцдаг.

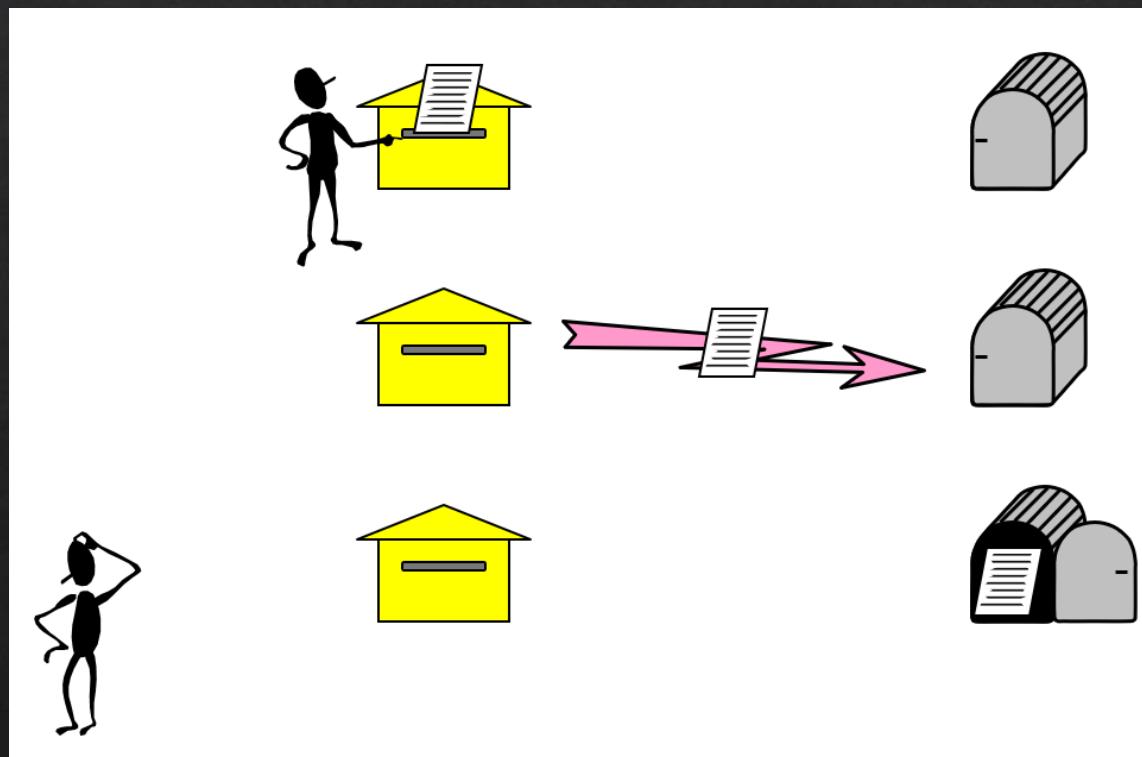
Synchronous Sends

- ❖ Илгээгч нь мессеж хүлээн авсан тухай мэдээлэл авдаг.
- ❖ Факсын хонх эсвэл “Ок”-хуудастай адил.



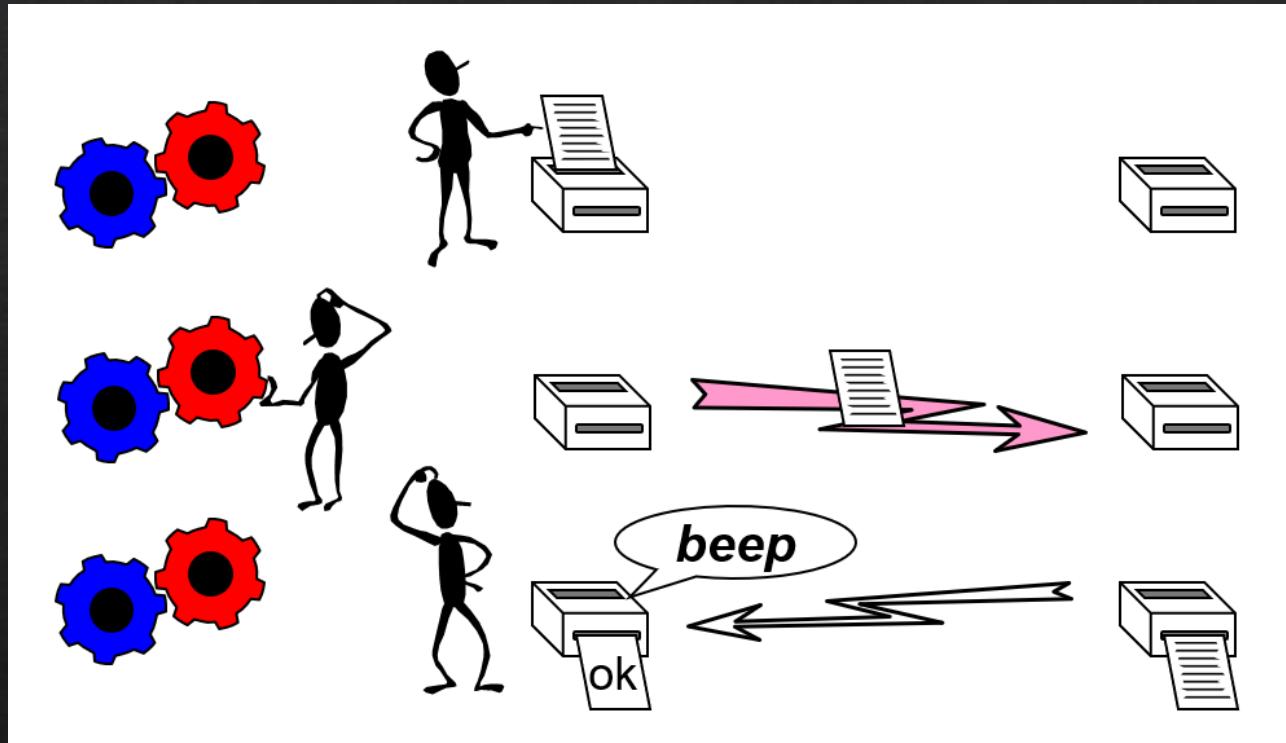
Buffered = Asynchronous Sends

- ❖ Мессеж илгээж буйгаа л мэддэг.



Non-Blocking Operations

- ❖ Non-Blocking үйлдэл нь шууд буцдаг. Ингэснээр тухайн дэд програм бусад ажлаа гүйцэтгэх боломжийг олгодог.



Мессеж

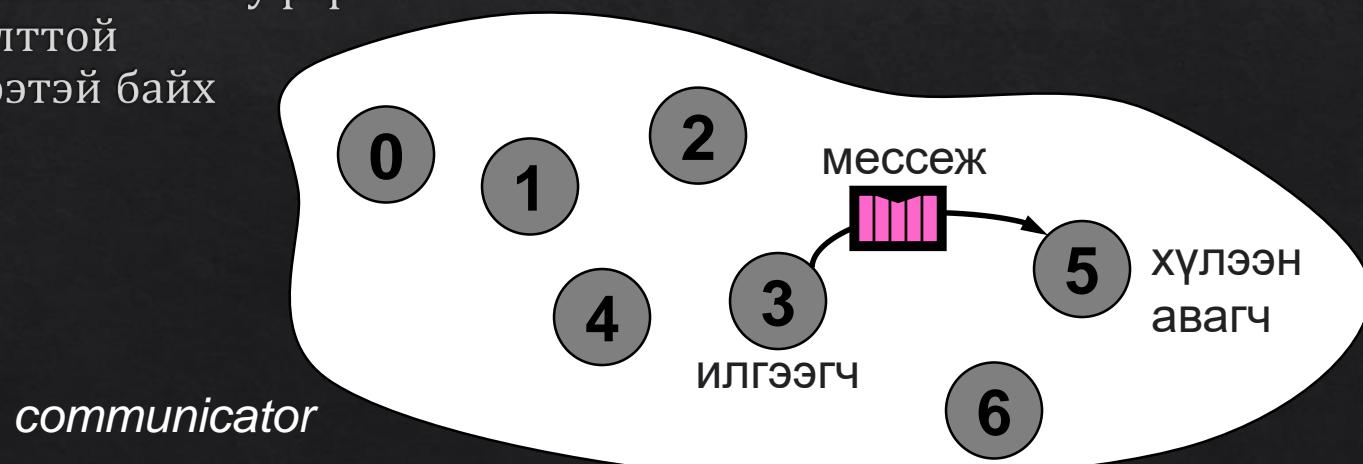
- ❖ Мессеж нь янз бүрийн өгөгдлийн төрөлтэй хэд хэдэн элементийг агуулж болно.
- ❖ MPI datatype:
 - ❖ Үндсэн datatype
 - ❖ Үүсмэл datatype 
- ❖ Datatype нь санах ойд байрлаж буй өгөгдлийн төрлийг нийтлэг болгоход хэрэглэгддэг.

2345	654	96574	-12	7676
------	-----	-------	-----	------

Жишээ: 5 ширхэг бүхэл тоо агуулсан мессеж

Point-to-Point харилцаа

- ❖ *Communicator* доторх нэг процесс нөгөө рүү мессеж илгээдэг.
- ❖ Мэдээлэл солилцоо нь дараах шаардлагуудыг хангах ёстой:
 - ❖ Нэг нь нөгөөгийхөө *rank*-ийг зөв тодорхойлсон байх
 - ❖ Ижил *communicator*-т байх бөгөөд *tag* болон *мессеж өгөгдлийн төрөл* харилцан тохирох
 - ❖ Хүлээн авагчийн буфер хангалттой хэмжээтэй байх



MPI_ANY_SOURCE ба MPI_ANY_TAG

- ❖ Бүх илгээгчээс хүлээн авах — source = **MPI_ANY_SOURCE**,
- ❖ Ямар ч tag-ийг хүлээн авах — tag = **MPI_ANY_TAG**
- ❖ Илгээгчдийн мэдээлэл хүлээн авагчийн **status** параметерт орж ирнэ.
- ❖ **status.MPI_SOURCE**
status.MPI_TAG
- ❖ **int MPI_Get_count(MPI_Status *status,
MPI_Datatype datatype);**

МЭДЭЭЛЭЛ СОЛИЛЦООНЫ ГОРИМ

❖ Илгээх горим

Synchronous send MPI _ SSEND	Зөвхөн хүлээн авалт эхэлсний дараа дуусна.
Buffered [asynchronous] send MPI _ BSEND	Хүлээн авагчаас хамааралгүй (алдаа гараагүй бол) шууд дуусна. (MPI _BUFFER _ATTACH)
Synchronous MPI _ SEND	Стандарт илгээлт. MPI _BUFFER _ATTACH болон дотоод буфер хэрэглэх боломжтой.
Ready send MPI _ RSEND	Зөвхөн хүлээн авалт бэлэн болсон үед л эхлэнэ. <i>Маш аюултай!</i>

❖ Бүх горимыг хүлээн авах

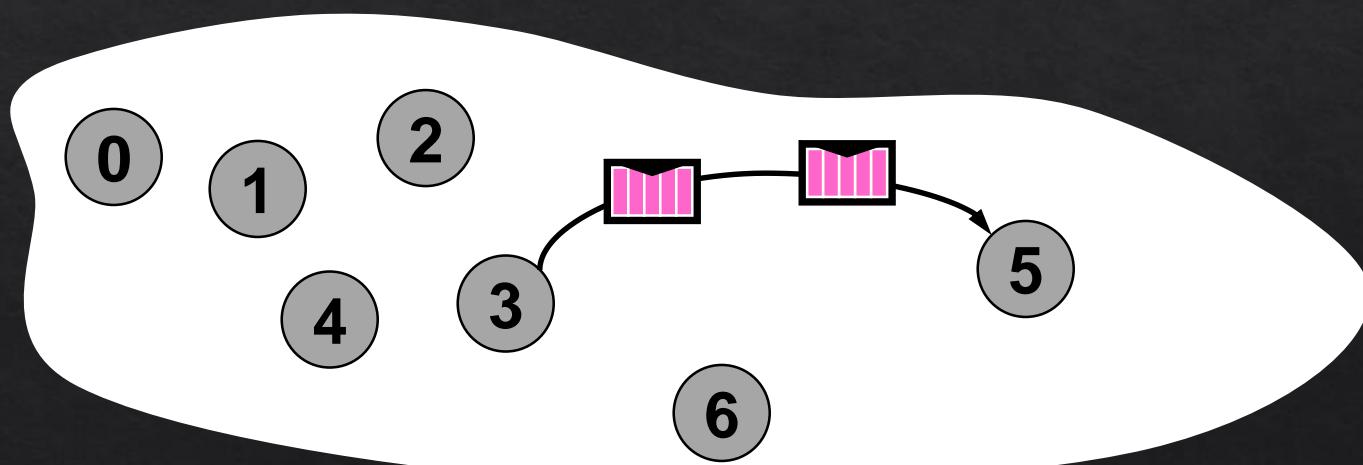
Receive MPI _ RECV	Мессеж (өгөгдөл) бүрэн ирсний дараа дуусна.
------------------------------	---

МЭДЭЭЛЭЛ СОЛИЛЦООНЫ ГОРИМ

- ❖ Standard send (**MPI_SEND**)
 - ❖ minimal transfer time
 - ❖ may block due to synchronous mode
 - ❖ —> risks with synchronous send
- ❖ Synchronous send (**MPI_SSEND**)
 - ❖ risks of deadlock, serialization and waiting (idle time)
 - ❖ high latency / best bandwidth
- ❖ Buffered send (**MPI_BSEND**)
 - ❖ low latency / bad bandwidth
- ❖ Ready send (**MPI_RSEND**)
 - ❖ use **never**, except you have a 200% guarantee that Recv is already called in the current version and all future versions of your code

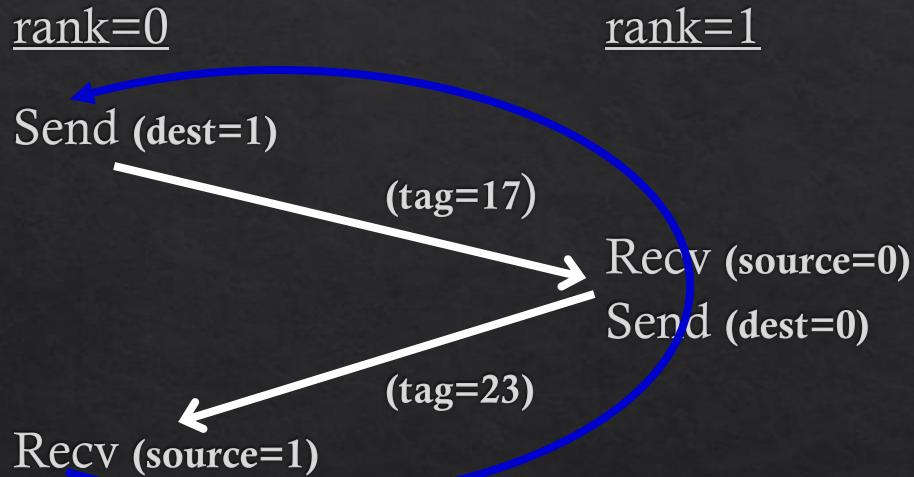
Мессеж дараалал хадгалалт

- ❖ Ижил холболт дээрх мессежүүд дарааллаа хадгална. Жнь: ижил *communicator*, илгээгч болон хүлээн авагчийн *rank*;
- ❖ Мессежүүд нь бие биенээ гүйцэж түрүүлэхгүй.
- ❖ Энэ зарчим non-synchronous send-д адилхан.

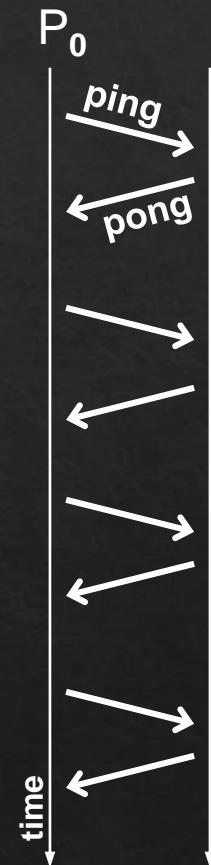


- ❖ Хэрэв мессежийн шалгуурууд нь ижил бол хүлээн авагчдийн хувьд ч мөн дараалал нь хадгалагдана.

Жишээ — Ping pong



```
if (my_rank==0)
    MPI_Send( ... dest=1 ... )
    MPI_Recv( ... source=1 ... )
else
    MPI_Recv( ... source=0 ... )
    MPI_Send( ... dest=0 ... )
fi
```



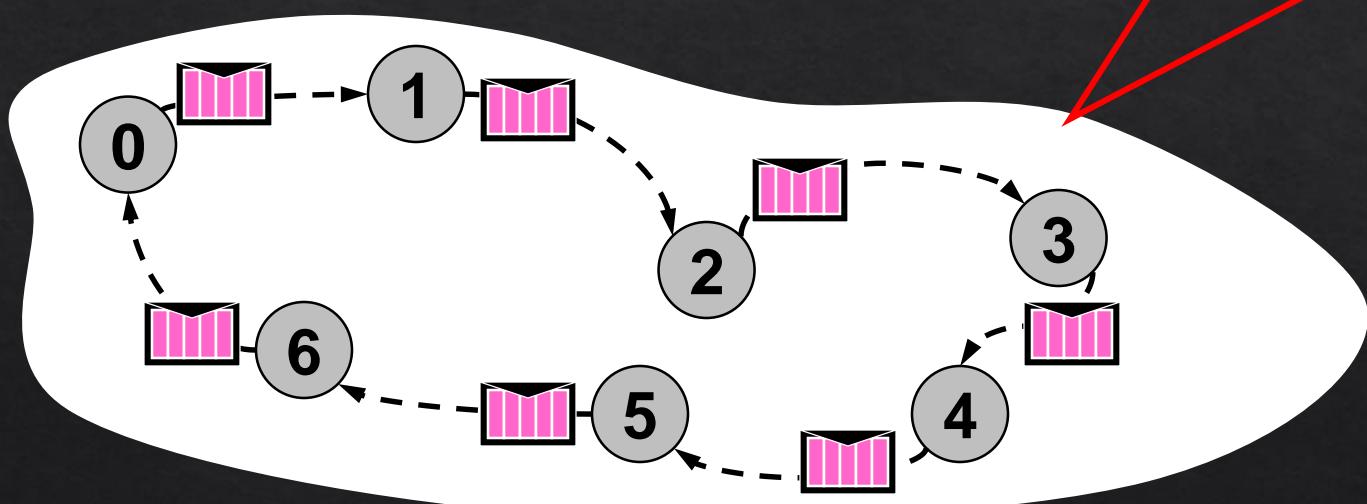
Deadlock

❖ MPI процесс бүр:

`MPI_Ssend(..., right_rank, ...)`

`MPI_Recv(..., left_rank, ...)`

MPI_Recv нь баруун гар талын MPI процесстоо дуудагдах боломжгүй тул хүлээнэ, хэзээ ч буцахгүй.

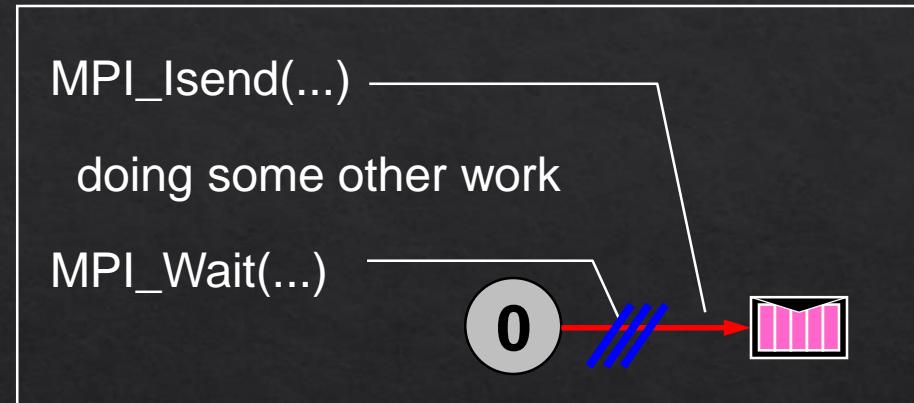


Non-Blocking Communication

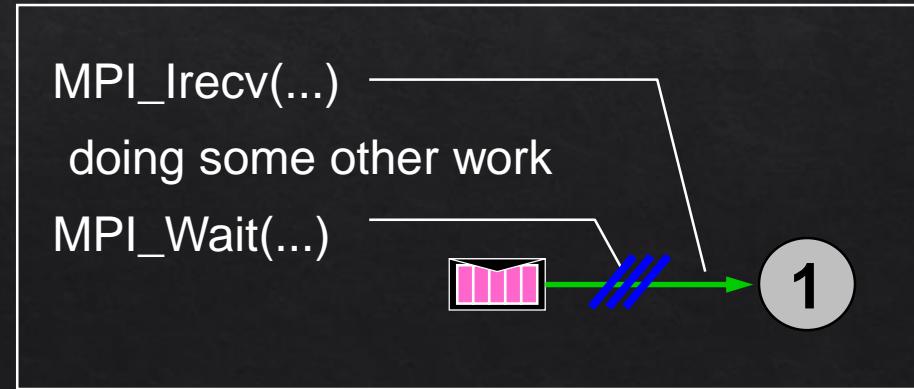
- ❖ idle time болон deadlocks-oos зайлсхийх
- ❖ Non-Blocking мэдээлэл солилцооны алхмыг 3 үе шатад хуваана
 - ❖ Non-Blocking мэдээлэл солилцоог бэлтгэх
 - ❖ Шууд буцдаг (Immediately)
 - ❖ Цикл нь MPI_I... гэж эхлэнэ.
 - ❖ θөр ажил гүйцэтгэх
 - ❖ “Latency hiding”
 - ❖ Non-Blocking мэдээлэл солилцоо гүйцэтгэгдэхийг хүлээх

Non-Blocking жишиээ

- Non-blocking send



- Non-blocking receive



= waiting until operation locally completed

Request Handles

- ❖ Локал хувьсагчид хадгалагдах ёстой: **MPI_Request**
- ❖ Утга нь:
 - ❖ non-blocking мэдээлэл солилцооны ***циклээр үүсгэгдэнэ***
 - ❖ **MPI_WAIT** хэсэгт ***хэрэглэгддэг (гарч ирнэ)***.

Thanks.