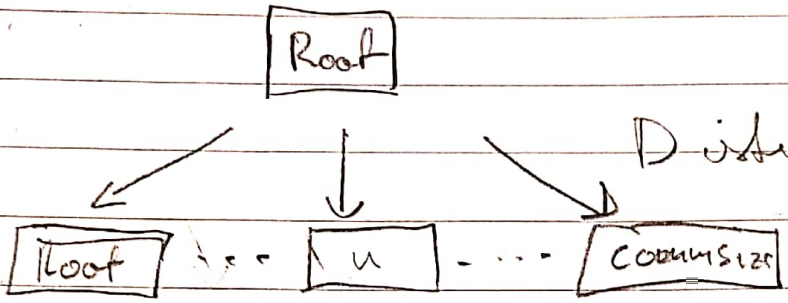
2/. Take 4 processes with 2 iterations and and Laplace 7 kernel.

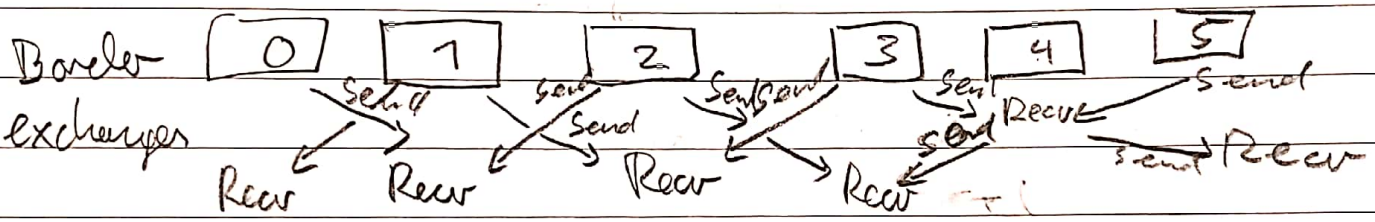
1/ first, the image -> rawdata is scattered with scatterv to all processes.

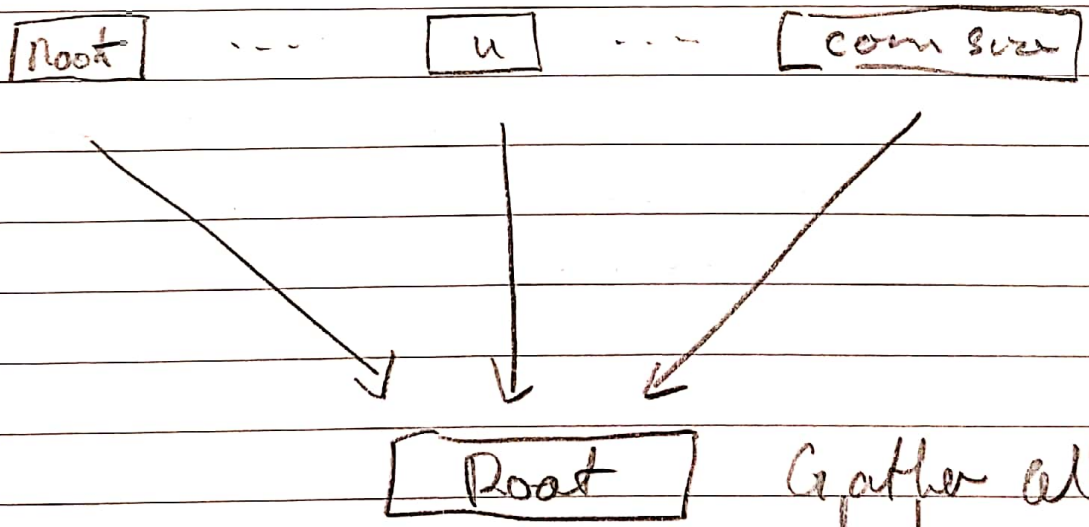
Then for each rank, in this case for 4 ranks borders are exchanged two times, not for 0 and the last (they only have one border).
That is a total of 4+2 = 6 exchanges over 2 iterations, which means 12 exchanges.

Last, the subimages are gathered at scatterv.



for (i < iterations)





2) Sizeof(Pixel) = 3 Bytes.

1. Scatter distributes 2334 × 4000 pixels

2. for each iteration, each process except for 0 and 4 sends 2334 pixels
In total $2 \cdot 2 \cdot 2334 + 2 \cdot 2334$ pixels

3. We then gather the image where each sub-image sends its partition, in total
2334 × 4000 pixels

2 iterations so the second stage is doubled and we have

$$n = 2334 \times 4000 + 2(2 \cdot 2334 + 2 \cdot 2334) + 2334 \times 4000$$
$$= 18,720000 \text{ Mpixels}$$
$$\times 3$$
$$= 56,160024 \text{ Bytes}$$
$$\simeq \underline{56.16 \text{ MB}}$$

3) The time spent in communications for 8 would be proportional to the amount additional data transfered We would then have 6 instead of 2 ~~weights~~ variables distributing to two neighbours Following the same calculation as before we then have

$n_2 \triangleq 56,35\,MB$

$= 5[illegible]3$

$\frac{n_2}{n_1} = \frac{1,00338191}{} \triangleq 0,3\%$ more

Not much bigger, the "heaviest" operations are the initial sends.

However if we just think about the border exchanges we end up sending 2,33 times more data.

4) See excel picture appear to approach a speed of roughly 2.6 ~~8~~.

iterations:	runtime:	iterations:	runtime:	speedup:
1	0,185	1	0,097	1,907216
5	0,886	5	0,365	2,427397
20	3,558	20	1,393	2,5542
50	9,13	50	3,504	2,605594
100	18,388	100	7,034	2,61416
200	37,671	200	12,305	3,061438
300	56,082	300	21,206	2,644629
400	75,827	400	28,356	2,674108
500	94,465	500	35,349	2,672353
700	133,595	700	49,642	2,691169
900	171,086	900	63,85	2,679499
1000	190,071	1000	70,969	2,678226

COMM size used for all: 6