10.4 Fox et al. (1998) present a method of compare and exchange in which individual numbers are exchanged between the processors, say P_0 and P_1. Suppose each processor has four numbers in its group. Processor P_0 sends the largest number in its group to processor P_1, while processor P_1 sends the smaller number of its group to processor P_0. Each processor inserts the retrieved number into its group, so that both still have n/p numbers. These actions are repeated with the new largest and smaller numbers. The algorithm is most conveniently implemented using a queue with a pointer to top or bottom. The algorithm can terminate when the insertions would be at the top and bottom of the groups. Write a parallel program to implement Fox' s method and evaluate it compared to the method described in the text.

```
while(true)
{
	isEven = (nPes%2);
	if(isEven)
	{
		send(&P0list.top(), P1);
		recv(&recValue, P1);
		if(P0list.top() > recValue)
		{
			P0list.pop();
			P0list.push(recValue);
		}
		else
			break;
	}
	else
	{
		recv(&recValue, P0);
		send(&P1list.top(), P0);
		if(P1list.top() < recValue)
		{
			P1list.pop();
			P1list.push(recValue);
		}
		else
			break;
		}
	}
```

10.5 Determine whether this code is correct, and if not, correct any mistakes.

```
evenprocess = (i % 2 == 0);
evenphase = 1;
for(step = 0; step < n; step++, evenphase = !evenphase){
        if((evenphase && evenprocess) || (!evenphase && !evenprocess)){
                if(i != Pi + 1)
                {
                        send(&a, Pi + 1);
                        recv(&xm Pi + 1);
                        if(x < a) a = x;
                }
        } else {
                if(i > 0)
                {
                        recv(&x, Pi - 1);
                        send(&a, Pi - 1);
                        if(x > a) a = x;
                }
        }
}
```

Las correcciones hechas al código se ven en las dos condiciones agregadas. La primera que verifica que el primer proceso nunca vaya a pedir intercambio a su izquierda. La segunda, de igual manera, verifica que el último proceso no pida intercambio hacia su derecha.

10.6 The odd-even transposition sort of Section 10.2.2 was described on the assumption of an even number of numbers and processors. What changes to the code would be necessary to allow an odd number of numbers and processors?

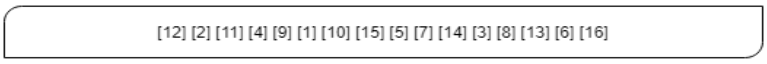
En el caso de la fase impar, la condición $if(i \leq n-3)$ cambiaría a la siguiente: $if(i \leq (n-3)+(n\%2))$. Por otro lado, para las fases pares se debe agregar una nueva condición, la cual controlará la recepción y el envío de mensajes con su proceso de la derecha, quedando de esta manera:

$$
\begin{aligned}
&if(i\,!=n-1)\,\{\\
&\quad recv(\&A, P_{i+1});\\
&\quad send(\&B, P_{i+1});\\
&\}
\end{aligned}
$$

10.12 Draw the compare-and-exchange circuit configuration for the odd-even mergesort algorithm described in Section 10.2.7 to sort 16 numbers. Sort the following sequence by hand using the odd-even mergesort algorithm.

Paso 1

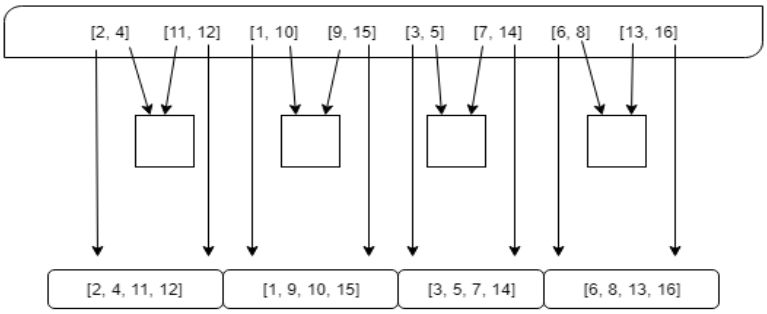
Todos los procesos tienen un solo elemento de la lista de números.



Paso 2

Paso 3

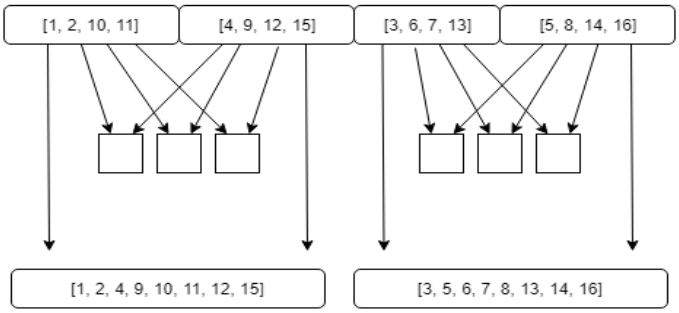
Se colocan los elementos de posiciones pares en un solo lado y los elementos de posiciones impares del otro



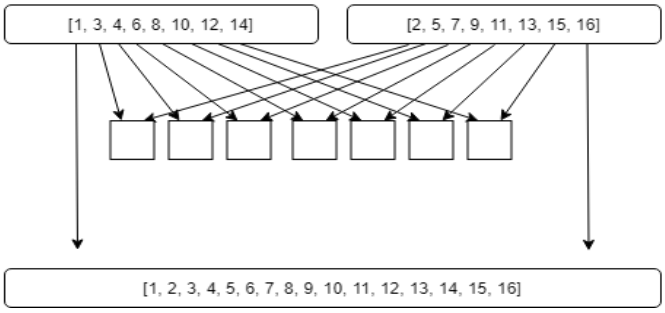
Paso 4

Compare and exchange

Repetir Paso 3 y 4



Repetir Paso 3 y 4



10.14 Compare Batcher' s odd-even mergesort algorithm(Section 10.2.5_ and his bitonic mergesort algorithm (Section 10.2.6), and asses their relative advantages for parallel implementation on a message-passing multicomputer.

En cuanto a lo que tiempo de complejidad se refiere, se puede decir que ambos algoritmos son equivalentes, debido a que los dos pertenecen a $O(\log^2 n)$ para cuando se tienen n procesos. Las diferencias vienen en cómo funcionan internamente, ya que el bitonic mergesort está basado en las secuencias bitónicas, las cuales se caracterizan por ser secuencias numéricas que crecen monotónicamente hasta un valor máximo y, a partir de ahí, se convierte en una secuencia monotónicamente decreciente. La ventaja que tiene el bitonic mergesort es que puede ser mapeado a redes de malla o hipercubos.