11.4.1.1 Blocking and Buffered Sends

The default means of sending point-to-point messages with `Send` and `Recv` represents one combination in a spectrum of available communication protocols. Both functions are known as *blocking* functions because they do not allow the execution of the program to continue until it is safe to do so. The `Send` method not only guarantees that it will not change the contents of the data buffer, but that any subsequent changes to the data buffer will not affect the message that is being sent. So if computation is allowed to proceed from a `Send` call it either means that the message has already been delivered or that the data has been copied into another buffer ready for delivery.

The default `Send` is a compromise between the safety of waiting to be sure that a message has been delivered and the efficiency of getting on with other tasks after sending the message immediately. The other send functions have similar function prototypes, but slightly different names. We briefly describe these send functions below: the interested reader should consult a dedicated MPI programming book [9, 10] for more details.

- The very safest, but possibly most inefficient, means of sending a message is to use a *blocking* synchronous send, `Ssend`. This function guarantees not to continue until the message has been delivered. This is a little like delivering a message by telephone conversation, because we cannot get on with our lives until the call has been made and the information has been relayed.
- A slightly more configurable version is `Bsend`, the *buffered* send. Like the plain `Send`, it allows the program to continue when safe, but this may happen faster since the message is copied to a separate buffer. This buffer must be supplied and configured by the user.
- At the top end of the spectrum, the most efficient means of sending a message is the *immediate* send `Isend`, which returns control to the program immediately, whether the message has been delivered, buffered or not yet acted on. This is a little like communicating via SMS text message in which we are able to press "send" and get on with other things safe in the knowledge that the recipient will get the information some time soon. Because it may be dangerous to overwrite the original data contained in the message, MPI provides functions for testing whether or not the message has been delivered. The `Isend` command gives back a handle (called an `MPI::Request`) which has a `Wait` method: this method instructs execution to "wait here" until the message has been sent.
- There are a few other flavours of `Send` including compatible combinations: an immediate send can make use of a user-supplied buffer using the buffered non-blocking combination `Ibsend`.

The default `Recv` function is also one of a spectrum of functions. It is technically a *blocking* function, because execution cannot continue until a suitable message has been received. There is also a *non blocking* immediate receive `Irecv` together with some utilities for probing whether there are any queued messages which match certain sources or tags. This means that your program, rather than waiting for messages to be received, could get on with useful work, occasionally going back to check for new information.