2 Answers

active oldest **votes**

1

Writes to a pipe up to the size of `PIPE_BUF` are atomic (included in `limits.h`), therefore you can easily pack your information into some type of struct, and write that to the pipe in your child process for the parent process to read. For instance, you could setup your struct to look like:

```
struct message
{
    int word_freq;
    char word[256];
};
```

Then simply do a read from your pipe with a buffer that is equal to `sizeof(struct message)` . That being said, keep in mind that it is best to only have either a single reader/writer to the pipe, or you can have multiple writers (because writes are atomic), but again, only a single reader. While multiple readers can be managed with pipes, the fact that reads are *not* atomic means that you could end up with scenarios where messages either get missed due to the non-deterministic nature of process scheduling, or you get garbled messages because a process doesn't complete a read and leaves part of a message in the pipe.

link

edited **2 mins ago**

answered **7 mins ago**

Jason

7,495 ●2 ●14