Vectored I/O

From Wikipedia, the free encyclopedia

> *This article is about the I/O method. For the vector addressing type, see Gather-scatter (vector addressing).*

In computing, **vectored I/O**, also known as **scatter/gather I/O**, is a method of input and output by which a single procedure-call sequentially writes data from multiple buffers to a single data stream or reads data from a data stream to multiple buffers. The buffers are given in a vector of buffers. *Scatter/gather* refers to the process of gathering data from, or scattering data into, the given set of buffers. Vectored I/O can operate synchronously or asynchronously. The main reasons for using vectored I/O are efficiency and convenience.

There are several usages for vectored I/O:

- Atomicity: if the particular vectored I/O implementation supports atomicity, a process can write into or read from a set of buffers to or from a file without risk that another thread or process might perform I/O on the same file between the first process' reads or writes, thereby corrupting the file or compromising the integrity of the input
- Concatenating output: an application that wants to write non-sequentially placed data in memory can do so in one vectored I/O operation. For example, writing a fixed-size header and its associated payload data that are placed non-sequentially in memory can be done by a single vectored I/O operation without first concatenating the header and the payload to another buffer
- Efficiency: one vectored I/O read or write can replace many ordinary reads or writes, and thus save on the overhead involved in syscalls
- Splitting input: when reading data that is in a format that defines a fixed-size header, one can use a vector of buffers in which the first buffer is the size of that header; and the second buffer will contain the data associated with the header

Standards bodies document the applicable functions `readv`[1] and `writev`[2] in POSIX 1003.1-2001 and the Single UNIX Specification version 2. The Windows API has analogous functions `ReadFileScatter` and `WriteFileGather`; however, unlike the POSIX functions, they require the alignment of each buffer on a memory page.[3] Windows Sockets provide separate `WSASend` and `WSARecv` functions without this requirement.

While working directly with a vector of buffers can be significantly harder than working with a single buffer, there are often higher-level APIs[4] for working efficiently that can mitigate the problem.

Examples

The following example prints "Hello Wikipedia Community!" to the standard output. Each word is saved into a single buffer and with only one call to `writev()`, all buffers are printed to the standard output.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include <sys/types.h>
#include <unistd.h>
#include <sys/uio.h>

#define NUMBUFS 3

int
main(int argc, char *argv[])
```

```
{
        const char *buf1 = "Hello ";
        const char *buf2 = "Wikipedia ";
        const char *buf3 = "Community!\n";

        struct iovec bufs[NUMBUFS];

        bufs[0].iov_base = (void*) buf1;
        bufs[0].iov_len = strlen(buf1);

        bufs[1].iov_base = (void*) buf2;
        bufs[1].iov_len = strlen(buf2);

        bufs[2].iov_base = (void*) buf3;
        bufs[2].iov_len = strlen(buf3);

        if (-1 == writev(STDOUT_FILENO, bufs, NUMBUFS))
        {
                perror("writev()");
                exit(EXIT_FAILURE);
        }

        return 0;
}
```

References

1. readv (http://www.opengroup.org/onlinepubs/009695399/functions/readv.html) in the Single Unix Specification
2. writev (http://www.opengroup.org/onlinepubs/009695399/functions/writev.html) in the Single Unix Specification
3. ReadFileScatter (http://msdn2.microsoft.com/en-us/library/aa365469.aspx) in MSDN Library
4. Vstr (http://www.and.org/vstr/) the Vectored String API

Retrieved from "https://en.wikipedia.org/w/index.php?title=Vectored_I/O&oldid=648137883"

Categories: Input/output

- This page was last modified on 21 February 2015, at 03:56.
- Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. By using this site, you agree to the Terms of Use and Privacy Policy. Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.