CHINA LINUX KERNEL
中国Linux内核开发者大会

華中科技大學
网络空间安全学院
School of Cyber Science and Engineering, HUST

第19届中国Linux内核开发者大会

赞助单位

华中科技大学 | HUAWEI | 阿里云 | FUJITSU | 腾讯云
oppo | 火山引擎 | vivo | 龙芯中科 LOONGSON TECHNOLOGY | 蚂蚁集团 ANT GROUP

支持单位

清华大学 Tsinghua University | 迪捷软件 DIGIPROTO | OpenEuler | OpenAnolis 龙蜥社区 | OpenCloudOS

支持社区&媒体

CSDN | 云巅论剑 | sf 思否 | 阅码场 yomocode | InfoQ 极客传媒
51CTO | 开源江湖

2024年10月　湖北 • 武汉

華中科技大學

f2fs : large folio 特性的探索和实践

张细锐 vivo 存储系统工程师

荣乾锋 vivo 内存管理工程师

目录

Part One

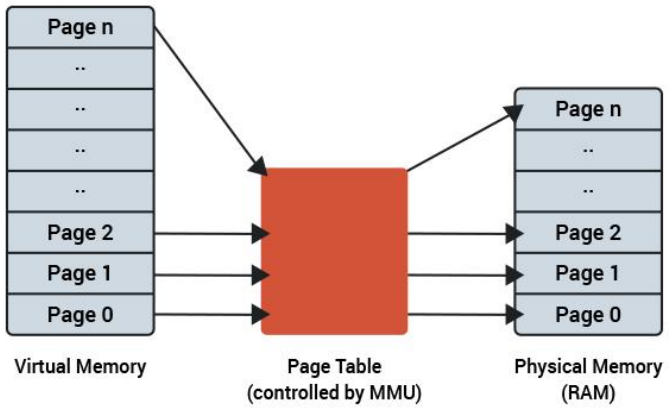
Memory folio 背景

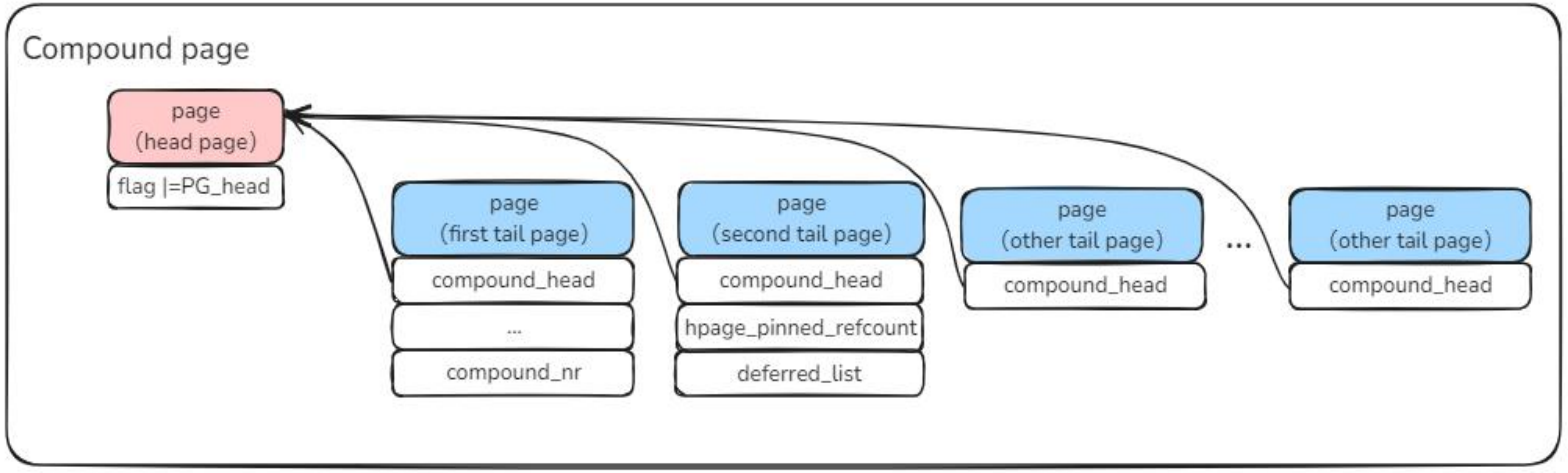
Memory folio 背景

page语义拓展

内存是以page管理单位，有些情况需要使用连续的大页内存，为了解决这个需求，引入了compound page概念，Compound page是对page进行了语义拓展

- 拓展前：page就表示单页(order-0 page)
- 拓展后：page可能表示单页(orde-0 page)、Compound page的head page，Compound page的tail page





Memory folio 背景

page多语义问题

- Page多语义导致page处理函数需要冗余判断
- 引入folio是为了解决page多语义导致的冗余判断问题

```
void end_page_writeback(struct page *page)
{
        if (PageReclaim(page)) {                //1.PageRclaim() 需要调用compound_head(page)拿到head page
                ClearPageReclaim(page);         //2.ClearPageReclaim() 需要调用compound_head(page)拿到head page
                rotate_reclaimable_page(page);
        }
        get_page(page);                         //3.get_page() 需要调用compound_head(page)拿到head page
        if (!test_clear_page_writeback(page))
                BUG();

        smp_mb__after_atomic();
        wake_up_page(page, PG_writeback);
        put_page(page);                        //4.put_page() 需要调用compound_head(page)拿到head page
}
```

Memory folio 背景

page多语义问题解决方法

- 新增folio结构，folio是page的封装，folio可以是单页，也可以是复合页(large folio)，同时在语义上folio保证不是tail page
- 新增以folio为参数一系列page替换api，对folio进行处理，去掉冗余判断
- 对page接口进行替换改造，减少兼容时期page多语义影响

```
struct folio {
  union {
    struct {unsigned long flags; ...};
    struct page page;
  };
/*-----------------------------------------------------*/
  union {
    struct {unsigned long _flags_1;...};
    struct page __page_1;/*fillers for alignment*/
  };
/*-----------------------------------------------------*/
  union {
    struct {unsigned long _flags_2;...};
    struct {unsigned long _flags_2a;...};
    struct page __page_2;/*fillers for alignment*/
  };
};
```

1

```
#define folio_alloc(...) alloc_hooks(folio_alloc_noprof(__VA_ARGS__))
void folio_get(struct folio *folio)
void folio_put(struct folio *folio)
/*filemap.c*/
struct inode *folio_inode(struct folio *folio)
void folio_lock(struct folio *folio)
void folio_unlock(struct folio *folio)

/*page-writeback.c */
bool filemap_dirty_folio(struct address_space *mapping, struct folio *folio)
void folio_end_writeback(struct folio *folio)
size_t copy_folio_from_iter_atomic(struct folio *folio,
                size_t offset, size_t bytes, struct iov_iter *i)
bool folio_mark_dirty(struct folio *folio)
void folio_wait_writeback(struct folio *folio)
```

2

```
static inline void lock_page(struct page *page)
{
    struct folio *folio;
    might_sleep();

    folio = page_folio(page); // 对page的处理转换成folio的处理
    if (!folio_trylock(folio))
        __folio_lock(folio);
}
```

3

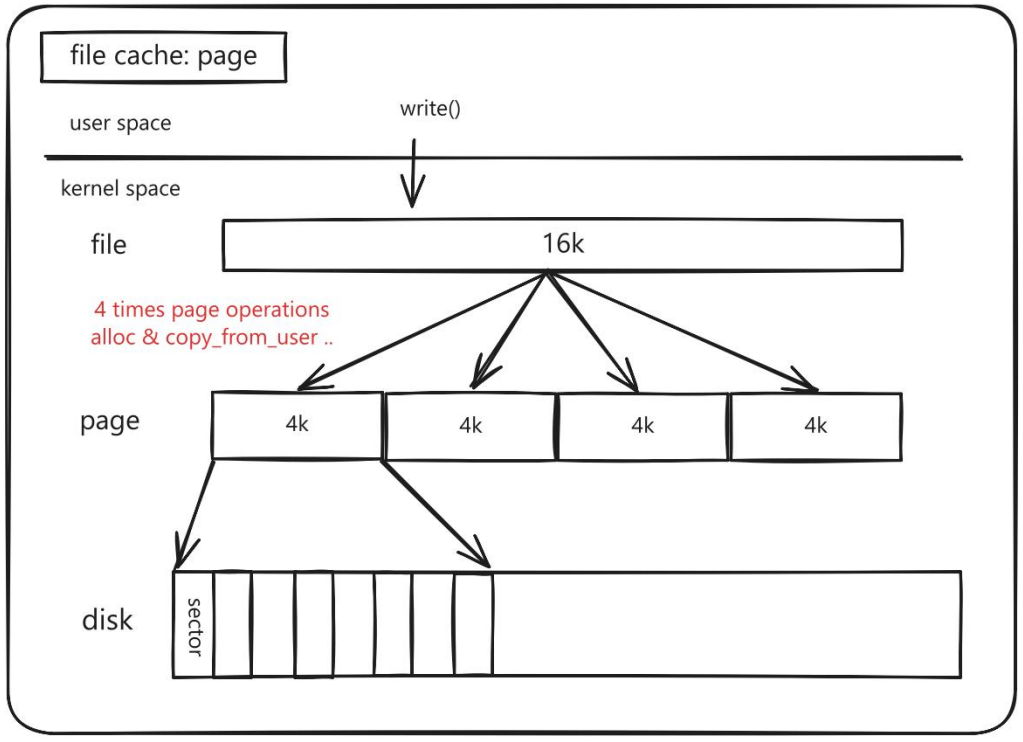
Part Two

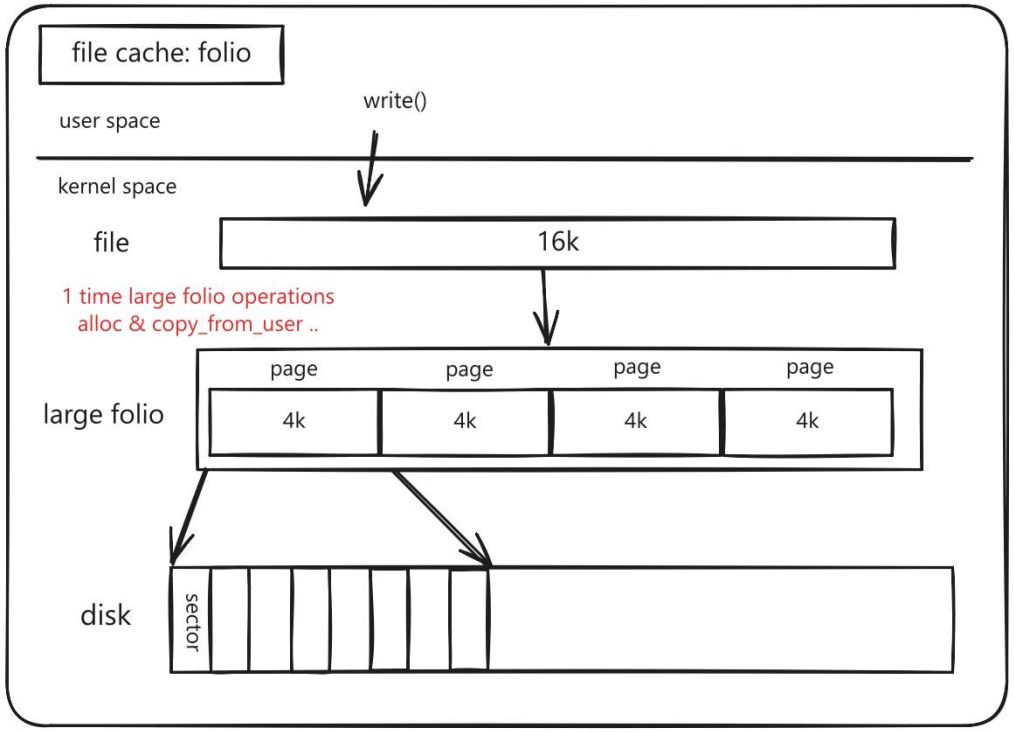
f2fs large folio探索和实践

f2fs large folio探索

folio的对文件系统影响

- 原理上讲，使用large pages可以提升文件系统buffer-io性能
- folio的引入为文件系统file cache机制使用large pages扫清障碍

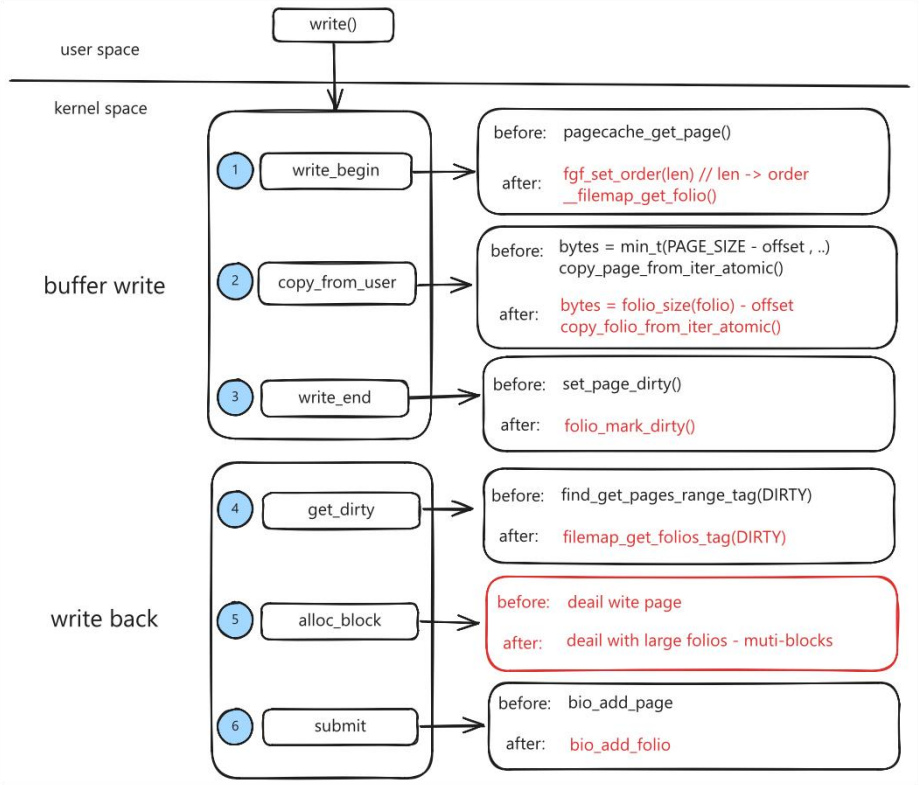




f2fs large folio探索 - 如何支持large folio

使用folio api支持large folio特性

- 从 page api切换到folio api
- 通过 mapping_set_large_folios(inode->i_mapping) 开启large folio

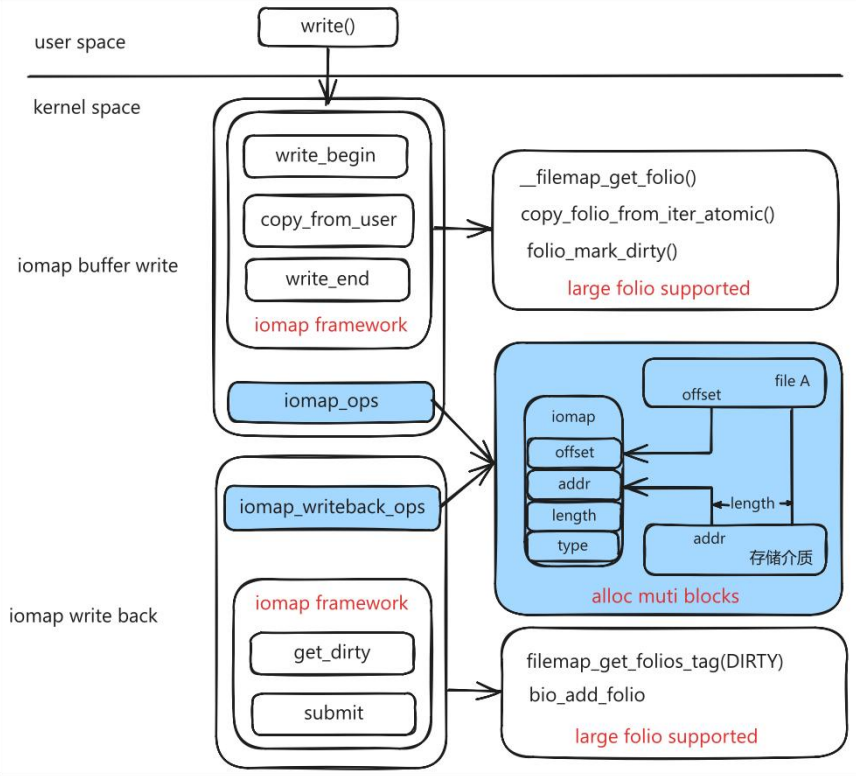



```
commit febc33cb352afb8c8dc87286635c35cc644fbdb9
Author: Pankaj Raghav <p.raghav@samsung.com>
Date:   Fri Jun 14 10:50:31 2024 +0000

    bcachefs: set fgf order hint before starting a buffered write

    This improves the buffered write performance up to 1.25x with default
    mount options and up to 1.57x when mounted with no_data_io option with
    the following fio workload:

    fio --name=bcachefs --filename=/mnt/test  --size=100G \
         --ioengine=io_uring --iodepth=16 --rw=write --bs=128k

    Signed-off-by: Pankaj Raghav <p.raghav@samsung.com>
    Signed-off-by: Kent Overstreet <kent.overstreet@linux.dev>

diff --git a/fs/bcachefs/fs-io-buffered.c b/fs/bcachefs/fs-io-buffered.c
index 865691dd0173..1355d618f988 100644
--- a/fs/bcachefs/fs-io-buffered.c
+++ b/fs/bcachefs/fs-io-buffered.c
@@ -677,7 +677,8 @@ int bch2_write_begin(struct file *file, struct address_space *mapping,

        bch2_pagecache_add_get(inode);

-       folio = __filemap_get_folio(mapping, pos >> PAGE_SHIFT, FGP_WRITEBEGIN,
+       folio = __filemap_get_folio(mapping, pos >> PAGE_SHIFT,
+                                   FGP_WRITEBEGIN | fgf_set_order(len),
                                    mapping_gfp_mask(mapping));
        if (IS_ERR_OR_NULL(folio))
                goto err_unlock;
@@ -819,7 +820,7 @@ static int __bch2_buffered_write(struct bch_inode_info *inode,
        darray_init(&fs);

        ret = bch2_filemap_get_contig_folios_d(mapping, pos, end,
-                                              FGP_WRITEBEGIN,
+                                              FGP_WRITEBEGIN | fgf_set_order(len),
                                               mapping_gfp_mask(mapping), &fs);
        if (ret)
                goto out;
```

f2fs large folio探索 - 如何支持large folio

使用iomap框架支持large folio特性

- 支持iomap框架，实现iomap hook函数
- 通过mapping_set_large_folios(inode->i_mapping)开启large folio

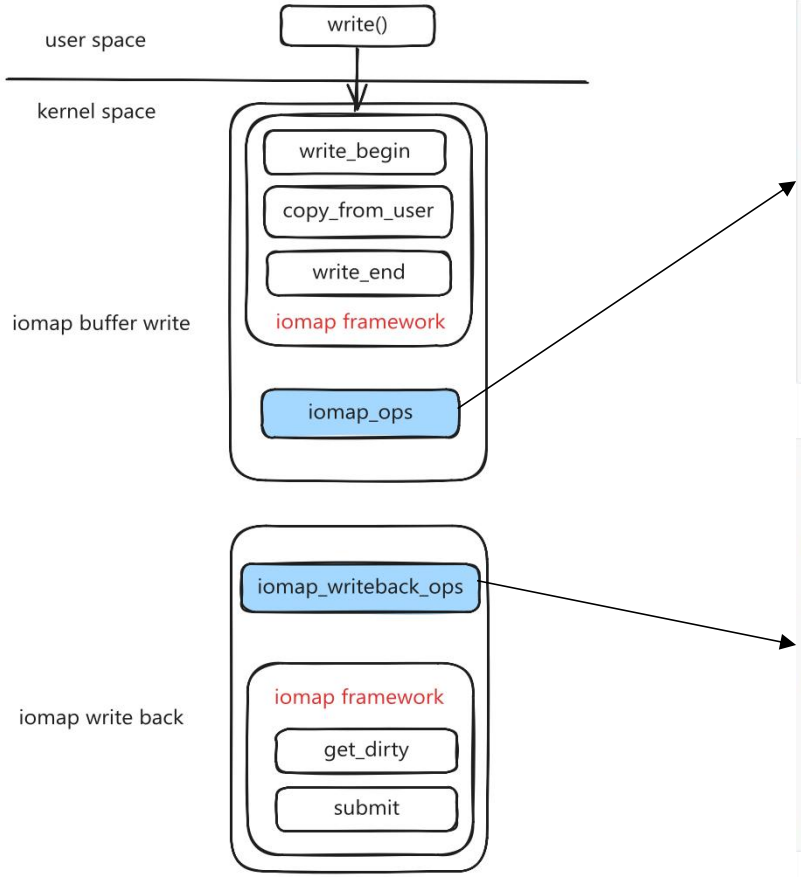



```
commit df2f9708ff1f23afdf6804bb16199e1903550582
Author: Johannes Thumshirn <johannes.thumshirn@wdc.com>
Date:   Tue May 14 00:37:18 2024 +0200

    zonefs: enable support for large folios

    Enable large folio support on zonefs.

    Cc: Matthew Wilcox <willy@infradead.org>
    Cc: Damien Le Moal <dlemoal@kernel.org>
    Signed-off-by: Johannes Thumshirn <johannes.thumshirn@wdc.com>
    Reviewed-by: Bill O'Donnell <bodonnel@redhat.com>
    Signed-off-by: Damien Le Moal <dlemoal@kernel.org>

diff --git a/fs/zonefs/super.c b/fs/zonefs/super.c
index 964fa7f24003..faf1eb87895d 100644
--- a/fs/zonefs/super.c
+++ b/fs/zonefs/super.c
@@ -662,6 +662,7 @@ static struct inode *zonefs_get_file_inode(struct inode *dir,
        inode->i_op = &zonefs_file_inode_operations;
        inode->i_fop = &zonefs_file_operations;
        inode->i_mapping->a_ops = &zonefs_file_aops;
+       mapping_set_large_folios(inode->i_mapping);

        /* Update the inode access rights depending on the zone condition */
        zonefs_inode_update_mode(inode);
```

f2fs large folio实践

基于iomap框架支持large folio

- 实现iomap_file_bufferd_write回调hook函数
- 实现iomap_writepags回调hook函数




```
const struct iomap_ops f2fs_buffered_write_iomap_ops = {
        .iomap_begin            = f2fs_buffered_write_iomap_begin,
        .iomap_end              = f2fs_buffered_write_iomap_end,
};

static ssize_t f2fs_buffered_write_iter(struct kiocb *iocb,
                                        struct iov_iter *from)
{
    ...
    /*使用iomap_file_buffer_write处理buffer write*/
    ret = iomap_file_buffered_write(iocb, from,
                        &f2fs_buffered_write_iomap_ops, NULL);
    ...
}
```

```
static const struct iomap_writeback_ops f2fs_writeback_ops = {
        .map_blocks = f2fs_iomap_map_blocks,
        ...
};

static int __f2fs_write_data_pages(struct address_space *mapping,
                                   struct writeback_control *wbc,
                                   enum iostat_type io_type)
{
    ...
    /*使用iomap_writepags处理回写流程*/
    ret = iomap_writepages(mapping, wbc, &wpc, &f2fs_writeback_ops);
    ...
}
```

f2fs large folio实践

Buffer write 对比测试结果

- 测试结果：64K buffer-write 性能提升 50.8%
- 测试命令：fio --name=f2fs --filename=/mnt/f2fs/test --size=8G --ioengine=io_uring --iodepth=16 --rw=write --bs=64k

Configuration	IOPS (avg)	BW (MB/s)	Latency (usec avg)	Write Duration (msec)
f2fs-large-folio-on	2755	181	5793	47571
f2fs-large-folio-off	1835	120	8699	71407

Part Three

未来展望

large folio未来展望

- Large folio 可以提升文件系统Buffer io性能，文件系统长期来看会趋于支持large folio，大家可以积极参与贡献开源提交
- 随着large folio的推广，large folio申请成功率优化，也会成为mm模块的一个热点方向

FS status (as of next-20240506)

fs	bh	page	large	wrpg	fs	bh	page	large	wrpg	fs	bh	page	large	wrpg
9p	n/a	netfs	no	yes	f2fs	no	no	no	no	ocfs2	no	no	no	yes
adfs	no	yes	no	yes	fat	no	yes	no	yes	omfs	no	yes	no	yes
affs	no	yes	no	yes	freevxfs	no	no	no	yes	orange	n/a	no	no	no
afs	n/a	netfs	yes	yes	fuse	n/a	no	no	yes	qnx4	no	yes	no	yes
bcachefs	yes	yes	yes	yes	gfs2	no	mostly	no	no	qnx6	no	no	no	yes
befs	no	yes	no	yes	hfs	no	no	no	yes	ramfs	yes	no	no	yes
bfs	no	yes	no	yes	hfsplus	no	no	no	yes	reiserfs	no	no	no	yes
btrfs	yes	no	no	yes	hostfs	yes	no	no	no	romfs	no	no	no	yes
ceph	n/a	no	no	no	hpfs	no	no	no	yes	smb	n/a	mostly	yes	yes
coda	n/a	yes	no	yes	hugetlb	yes	no	no	yes	squashfs	yes	no	no	yes
cramfs	yes	no	no	yes	isofs	no	no	no	yes	sysv	no	no	no	yes
ecryptfs	yes	no	no	no	jffs2	yes	no	no	yes	ubifs	yes	yes	no	yes
efs	no	no	no	yes	jfs	no	no	no	no	udf	no	yes	no	yes
erofs	yes	mostly	yes	yes	minix	no	no	no	yes	ufs	no	no	no	yes
exfat	no	yes	no	yes	nfs	n/a	mostly	no	yes	vboxfs	n/a	no	no	no
ext2	no	yes	no	yes	nilfs2	no	mostly	no	no	xfs	iomap	iomap	iomap	yes
ext4	no	mostly	no	yes	ntfs3	no	no	no	yes	zonefs	iomap	iomap	no	yes

文件系统large folio支持情况

Thank You~