Subject: [PATCH] Priority heap infrastructure enhancements
Posted by Balbir Singh on Wed, 18 Jun 2008 13:48:06 GMT
View Forum Message <> Reply to Message

This patch enhances the priority heap infrastructure to add a delete_max
routine. This patch and routines are helpful as

1. They allow me to delete nodes from the prio_heap (max heap), which is
   currently missing
2. This infrastructure would be useful for the soft limit patches I am working
   on for the memory controller

Some of the common code has been factored into heap_adjust() a.k.a heapify
in data structures terminology.

I am sending out this patch indepedent of the memory controller changes as
they deserve to be reviewed independently.

One limitation of the current heap_insert() routine is that it does not
insert an element which is greater than the root, when the heap slots
are fully used. I'll work on and review that interface and find a suitable
way to address that issue

I've tested them by porting the code to user space (very easy to do) and
I wrote a simple test routine, that ensures that elements are removed
from the heap in descending order.

Comments, Flames? Please do review closely!

Signed-off-by: Balbir Singh <balbir@linux.vnet.ibm.com>

```
---
 include/linux/prio_heap.h |   10 +++++++-
 lib/prio_heap.c           |   56 +++++++++++++++++++++++++++++++-------------
 2 files changed, 48 insertions(+), 18 deletions(-)

diff --git a/include/linux/prio_heap.h b/include/linux/prio_heap.h
index 0809435..a3578bd 100644
--- a/include/linux/prio_heap.h
+++ b/include/linux/prio_heap.h
@@ -53,6 +53,14 @@ void heap_free(struct ptr_heap *heap);
  */
 extern void *heap_insert(struct ptr_heap *heap, void *p);


-
+/**
+ * heap_delete_max - delete the maximum element from the top of the heap
+ * @heap: The heap to be operated upon
+ *
```

```
+ * The top of the heap is removed, the last element is moved to the
+ * top and the entire heap is adjusted, so that the largest element bubbles
+ * up again
+ */
+extern void *heap_delete_max(struct ptr_heap *heap);

 #endif /* _LINUX_PRIO_HEAP_H */
diff --git a/lib/prio_heap.c b/lib/prio_heap.c
index 471944a..4476bc9 100644
--- a/lib/prio_heap.c
+++ b/lib/prio_heap.c
@@ -23,11 +23,33 @@ void heap_free(struct ptr_heap *heap)
  kfree(heap->ptrs);
 }

+static void heap_adjust(struct ptr_heap *heap)
+{
+ int pos = 0;
+ void **ptrs = heap->ptrs;
+ void *p = ptrs[pos];
+
+ while (1) {
+  int left = 2 * pos + 1;
+  int right = 2 * pos + 2;
+  int largest = pos;
+  if (left < heap->size && heap->gt(ptrs[left], p))
+   largest = left;
+  if (right < heap->size && heap->gt(ptrs[right], ptrs[largest]))
+   largest = right;
+  if (largest == pos)
+   break;
+  /* Push p down the heap one level and bump one up */
+  ptrs[pos] = ptrs[largest];
+  ptrs[largest] = p;
+  pos = largest;
+ }
+}
+
 void *heap_insert(struct ptr_heap *heap, void *p)
 {
  void *res;
  void **ptrs = heap->ptrs;
- int pos;

  if (heap->size < heap->max) {
   /* Heap insertion */
@@ -49,22 +71,22 @@ void *heap_insert(struct ptr_heap *heap, void *p)
  /* Replace the current max and heapify */
```

```
  res = ptrs[0];
  ptrs[0] = p;
- pos = 0;
+ heap_adjust(heap);
+ return res;
+}
+
+void *heap_delete_max(struct ptr_heap *heap)
+{
+ void **ptrs = heap->ptrs;
+ void *res;
+
+ if (heap->size == 0)
+  return NULL;  /* The heap is empty */
+
+ res = ptrs[0];
+ heap->size--;
+ ptrs[0] = ptrs[heap->size]; /* Put a leaf on top */
+ heap_adjust(heap);

- while (1) {
-  int left = 2 * pos + 1;
-  int right = 2 * pos + 2;
-  int largest = pos;
-  if (left < heap->size && heap->gt(ptrs[left], p))
-   largest = left;
-  if (right < heap->size && heap->gt(ptrs[right], ptrs[largest]))
-   largest = right;
-  if (largest == pos)
-   break;
-  /* Push p down the heap one level and bump one up */
-  ptrs[pos] = ptrs[largest];
-  ptrs[largest] = p;
-  pos = largest;
- }
  return res;
 }
--
1.5.5.2
```

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: [PATCH] Priority heap infrastructure enhancements

Posted by Paul Menage on Sat, 21 Jun 2008 07:29:41 GMT
View Forum Message <> Reply to Message

On Wed, Jun 18, 2008 at 6:48 AM, Balbir Singh <balbir@linux.vnet.ibm.com> wrote:
>
> Some of the common code has been factored into heap_adjust() a.k.a heapify
> in data structures terminology.
>
> I am sending out this patch indepedent of the memory controller changes as
> they deserve to be reviewed independently.
>
> One limitation of the current heap_insert() routine is that it does not
> insert an element which is greater than the root, when the heap slots
> are fully used. I'll work on and review that interface and find a suitable
> way to address that issue

How else would you want it to behave? If you have a fixed size heap
and it's full, then you have to drop the largest value. (Well, you
could in theory drop the smallest value, but there's no quick way to
find that.)

>
> Comments, Flames? Please do review closely!
>
> Signed-off-by: Balbir Singh <balbir@linux.vnet.ibm.com>

Looks fine.

Reviewed-by: Paul Menage <menage@google.com>

Paul

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

Subject: Re: [PATCH] Priority heap infrastructure enhancements
Posted by Balbir Singh on Sat, 21 Jun 2008 08:05:38 GMT
View Forum Message <> Reply to Message

Paul Menage wrote:
> On Wed, Jun 18, 2008 at 6:48 AM, Balbir Singh <balbir@linux.vnet.ibm.com> wrote:
>> Some of the common code has been factored into heap_adjust() a.k.a heapify
>> in data structures terminology.
>>
>> I am sending out this patch indepedent of the memory controller changes as
>> they deserve to be reviewed independently.

>>
>> One limitation of the current heap_insert() routine is that it does not
>> insert an element which is greater than the root, when the heap slots
>> are fully used. I'll work on and review that interface and find a suitable
>> way to address that issue
>
> How else would you want it to behave? If you have a fixed size heap
> and it's full, then you have to drop the largest value. (Well, you
> could in theory drop the smallest value, but there's no quick way to
> find that.)
>

I would like to be able to drop the smallest value. Since we cannot drop the
smallest value, dropping a leaf (heap->size) should be sufficiently good enough.
I want a max heap and losing the root of the heap does not work for me.

>> Comments, Flames? Please do review closely!
>>
>> Signed-off-by: Balbir Singh <balbir@linux.vnet.ibm.com>
>
> Looks fine.
>
> Reviewed-by: Paul Menage <menage@google.com>

Thanks for the review!

--
Warm Regards,
Balbir Singh
Linux Technology Center
IBM, ISTL

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: [PATCH] Priority heap infrastructure enhancements
Posted by Paul Menage on Sat, 21 Jun 2008 08:11:24 GMT
View Forum Message <> Reply to Message

On Sat, Jun 21, 2008 at 1:05 AM, Balbir Singh <balbir@linux.vnet.ibm.com> wrote:
>
> I would like to be able to drop the smallest value. Since we cannot drop the
> smallest value, dropping a leaf (heap->size) should be sufficiently good enough.
> I want a max heap and losing the root of the heap does not work for me.
>

What are you actually trying to do? Can you get round this by just
inverting your "gt" operator? i.e. provide one that actually
implements "less-than"?

Paul

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: [PATCH] Priority heap infrastructure enhancements
Posted by Balbir Singh on Sat, 21 Jun 2008 08:54:07 GMT
View Forum Message <> Reply to Message

Paul Menage wrote:
> On Sat, Jun 21, 2008 at 1:05 AM, Balbir Singh <balbir@linux.vnet.ibm.com> wrote:
>> I would like to be able to drop the smallest value. Since we cannot drop the
>> smallest value, dropping a leaf (heap->size) should be sufficiently good enough.
>> I want a max heap and losing the root of the heap does not work for me.
>>
>
> What are you actually trying to do? Can you get round this by just
> inverting your "gt" operator? i.e. provide one that actually
> implements "less-than"?

Paul,

That would convert the entire heap to a min-heap. With my soft limit patches,
that I am working on for the memory controller; I want the controller that has
exceeded it's soft limit by the maximum amount to be picked off the heap, so
that we can reclaim from it on memory contention.

Ideally, I would also like to be able to find an existing node in the heap, but
that is hard. I can work around that problem for now.

--
Warm Regards,
Balbir Singh
Linux Technology Center
IBM, ISTL

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: [PATCH] Priority heap infrastructure enhancements
Posted by Paul Menage on Sat, 21 Jun 2008 16:03:44 GMT
View Forum Message <> Reply to Message

On Sat, Jun 21, 2008 at 1:05 AM, Balbir Singh <balbir@linux.vnet.ibm.com> wrote:
>
> I would like to be able to drop the smallest value. Since we cannot drop the
> smallest value, dropping a leaf (heap->size) should be sufficiently good enough.
> I want a max heap and losing the root of the heap does not work for me.

Dropping the last element will give you an "approximate-max" heap -
once you've finished building the heap, for a heap depth of N I think
the first N-1 calls to heap_delete_max() will give correct results,
but any time after that you could be missing things that were dropped
incorrectly during the building phase.

If this is OK for what you're trying to do, then either:

- adding a adding a heap_insert_drop_leaf_on_overflow() function, or

- adding a flag to the heap structure indicating what to do on overflow

sounds fine.

Paul

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

Subject: Re: [PATCH] Priority heap infrastructure enhancements
Posted by akpm on Tue, 01 Jul 2008 08:14:14 GMT
View Forum Message <> Reply to Message

On Sat, 21 Jun 2008 13:35:38 +0530 Balbir Singh <balbir@linux.vnet.ibm.com> wrote:

> Paul Menage wrote:
> > On Wed, Jun 18, 2008 at 6:48 AM, Balbir Singh <balbir@linux.vnet.ibm.com> wrote:
> >> Some of the common code has been factored into heap_adjust() a.k.a heapify
> >> in data structures terminology.
> >>
> >> I am sending out this patch indepedent of the memory controller changes as
> >> they deserve to be reviewed independently.
> >>
> >> One limitation of the current heap_insert() routine is that it does not
> >> insert an element which is greater than the root, when the heap slots
> >> are fully used. I'll work on and review that interface and find a suitable

> >> way to address that issue
> >
> > How else would you want it to behave? If you have a fixed size heap
> > and it's full, then you have to drop the largest value. (Well, you
> > could in theory drop the smallest value, but there's no quick way to
> > find that.)
> >
>
> I would like to be able to drop the smallest value. Since we cannot drop the
> smallest value, dropping a leaf (heap->size) should be sufficiently good enough.
> I want a max heap and losing the root of the heap does not work for me.
>
>
> >> Comments, Flames? Please do review closely!
> >>
> >> Signed-off-by: Balbir Singh <balbir@linux.vnet.ibm.com>
> >
> > Looks fine.
> >
> > Reviewed-by: Paul Menage <menage@google.com>
>
> Thanks for the review!
>

yup, thanks.

I'll duck the patch until we have some code which uses it. Please
retain Paul's Reviewed-by: for that occasion.

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: [PATCH] Priority heap infrastructure enhancements
Posted by Balbir Singh on Tue, 01 Jul 2008 08:33:40 GMT
View Forum Message <> Reply to Message

Andrew Morton wrote:

>>> Reviewed-by: Paul Menage <menage@google.com>
>> Thanks for the review!
>>
>
> yup, thanks.
>
> I'll duck the patch until we have some code which uses it. Please

> retain Paul's Reviewed-by: for that occasion.

Sure, I'll keep the reviewed-by around. I might move over to a different data structure for soft limits, but I don't know for sure yet.

--
Warm Regards,
Balbir Singh
Linux Technology Center
IBM, ISTL

_____

Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers