دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر

عنوان:

گزارش پروژه درسی

اعضای گروه

فرید فتوحی

حسین رمضانی یادگار

ابوالفضل فرهادی

نام درس

سیستم‌های عامل

نیم‌سال اول ۱۴۰۱-۱۴۰۲

نام استاد درس

دکتر اسدی

تعریف پروژه: در این پروژه هدف باز طراحی و پیاده سازی توابع تخصیص حافظه ی Heap بصورت dynamic میباشد.استفاده از این توابع معمولا در قسمت های از کد که اندازه ی یک رشته یا آرایه ای از یک دیتا استراکچر مشخص نیست یا اندازه ی آن به قدری بزرگ است که در حافظه ی stack جا نمیشود ، استفاده میشوند.لازم به ذکر است که این توابع feature اضافه تری نسبت به توابع استاندارد دارند که میتوان الگوریتم اختصاص دادن حافظه را در استفاده از این توابع تنظیم کرد و همچنین مقادیر addressable byte را نیز جایگزاری نمود.

۱ تابعmalloc

با توجه به تعریف تابع malloc ، میتوان با توجه به الگوریتم پیاده سازی شده ، بلاک مورد نظر در صورت وجود را پیدا کرد و split را در صورت نیاز انجام داد.اگر بلاک مورد نظری ک بتواند خواسته ی ورودی را اغناع کند وجود نداشته باشد باید پوینتر مربوط به فضای تخصیص یافته از heap را به مقدار لازم جا به جا نمود و سپس بلاک جدید را ساخت و به خروجی پاس داد.همچنین این تابع مقدار یک بایتی پاس داده شده تحت عنوان fill را در همه ی خانه های حافظه ی مورد نظر قرار میدهد.

۱-۱ تابع splitblock

این تابع با توجه به الگوریتم در نظر گرفته شده، مقدار حافظه ی از پیش تعیین شده ی بلاک را به دو قسمت که قسمت اول مورد استفاده ی تابع malloc قرار میگیرد به اندازه ی لازم درخواست شده است.اگر از الگوریتم first-fit استفاده شده باشد ، مقدار بلاکی که از بلاک اصلی جدا شده است حتی اگر یک متغیر ۳۲ بیتی را بتواند در خود جای دهد، بعنوان بلاک مستقل در نظر میگیریم و لینک لیست بلاک های مربوطه را update میکنیم.ولی اگر از الگوریتم buddy اسفتاده شود بلاک اصلی را بصورت درختی از اندازه ی توان ۲ میشکنیم و سپس مقدار درخواست شده را به آن قسمت که کافی است و لزوما fit نیست،map میکنیم.لازم به ذکر است که برای جلوگیری از سربار اضافی بقیه ی بلاک ها را باهم merge میکنیم.

۱-۲ تابع findblock

این توابع وظیفه ی search بین همه بلاک ها و پیدا کردن بلاک مناسب از طریق iteration بین بلاک لیست است. در صورت موجود بودن بلاکی برای اغناع خواسته ی ورودی، بلاک مورد نظر به تابع split پاس داده شده تا بقیه ی افعال روی آن اعمال شود.درصورتی که بلاک موجود نباشد،باید پوینتر مشخص کننده ی فضای تخصیصی heap را جا به جا نمود و این جابه جایی به اندازه ی "لازم" خواهد بود.وظیفه ی این جا به جایی را تابع extendheap بر عهده دارد.

۱-۳ تابع extendheap

این تابع همانطور که گفته شد بر حسب نیاز فضای قابل تخصیص heap را گسترش میدهد.در حقیقت pointer break را جا به جا میکند.

۲ تابع showstats

طبق خواسته ی پروژه ، بلاک های allocated و free را با فرمت مورد نظر چاپ میکند و در نهایت دیتای تجمعی از کل فضای اختصاص یافته و اختصاص نیافته و خالی (حفره) و همچنین تفاضل پوینتر break از کل فضا ی مورد استفاده ی heap (شروع list linked) را چاپ میکند.

۳ تابع free

این تابع در نقش آزاد کننده ی بلاک مورد نظر ظاهر میشود.علاوه تغییر flag مربوط به تخصیص یافتگی بلاک ، راه کاری که بصورت ضمنی برای جلوگیری از fragmentation را با تابع nofrag پیاده میکند. این راه کار به این صورت است که چک میشود تا بلاک های همجوار بلاک مورد نظر free هستند یا خیر اگر free باشند تابع nofrag بلاک اصلی و بلاک همجوار را با هم merge میکند.یعنی هر بلاک در صورت آزاد شدن اگر امکان ترکیب شدن با بلاک های همجوار خود را داشت ، این اتفاق میفتد و بلاک های خالی کنار یک دیگر به یک بلاک بزرگ تر که احتمال استفاده از آن بالا تر میرود،تبدیل میشود.

۳-۱ تابع nofrag

این تابع با توجه به آن چه ذکر شد در قسمت free ، دو بلاک همجوار را باهم merge میکند.لازم به ذکر است که این merge کردن صرفا جا به جایی pointer ها با توجه به سناریو ی مورد بحث، میباشد و در واقع هیچ عمل merge کردنی روی نمیدهد.

۴ تابع realloc

طبق تعریف استاندارد realloc ،این تابع میتواند در نقش توابع malloc و free علاوه بر عملکرد جداگانه ای که ارائه میکند هم ظاهر شود.با توجه به ورودی های داده شده به این توابع میتواند تبدیل شود و همان خروجی را بدهد.در صورتی که pointer پاس داده شده به تابع به یکی از بلاک های از قبل تخصیص داده شده یا خالی اشاره کند ، عملکرد اصلی realloc که تغییر سایز تخصیص یافته شده اتفاق میفتد.برای کاهش سایز موضوع خاصی پیش نمیاید و صرفا بلاک را split میکنیم.(باتوجه به الگوریتم مورد استفاده).اما در حالتی که بخواهیم سایز تخصیص یافته شده را افزایش دهیم،دو حالت پیش می آید.در حالت اول اگر بلاک بعدی خالی باشد بلاک اصلی با بلاک بعدی خود ترکیب میشود و تا جای ممکن افزایش سایز اتفاق میفتد.در حالت دوم نیز بلاک بعدی خالی نیست و نمیتوان in-place تغییر سایز را اعمال کرد.در این حالت باید به دنبال بلاک دیگری برای اغناع این درخواست باشیم.در این صورت محتوای بلاک قبلی که در ورودی داده شده بود را در بلاک جدید که میتواند خواسته را جواب دهد کپی میکنیم.اگر هیچ یک از این دو حالت روی ندهد، نمیتوان افزایش سایز را اعمال کرد و null بازگردانده میشود. در تابع استاندارد مقادیر ست شده روی بلاک اصلی روی بلاک جدید که در حالت دوم اتفاق میفتاد،کپی میشوند ولی درخواست پروژه ست کردن این مقادیر با مقدارfill میباشد.

References

مطالب تکمیلی

پیوست‌های خود را در صورت وجود می‌توانید در این قسمت قرار دهید.