ساختمان داده‌ها (۲۲۸۲۲)

مدرس: حسین بومری

[زمستان ۹۹]

سوال ۵: درخت فضایی

نگارنده: آئیریا محمدی

از ساختار داده kd-tree استفاده می‌کنیم.

ساخت اگر برای بعدها ترتیب قائل شویم، به طور چرخشی در نوبت هر بعد نقاط را نسبت به آن بعد مرتب می‌کنیم و میانه را پیدا می‌کنیم و در درخت درج می‌کنیم. نقاط بزرگ‌تر و کوچک‌تر از میانه (از میان نقاط مربوط به این زیر درخت) را در چپ و راست راس درج شده درج می کنیم.

مرتبه زمانی آن از $O(nlogn)$ خواهد بود چرا که در $logn$ طبقه تعداد نقاطمان نصف می‌شوند و در هر طبقه باید برای ۲^i دسته با اندازه $n/۲^i$ هرکدام در $O(n/۲^i)$ میانه پیدا شود و پس از آن بر روی اعضای طبقه حرکت می‌کنیم تا آن ها را به دو لیست بر اساس بزرگ‌تری یا کوچک تری از میانه تقسیم کنیم. که این دو فرایند خطی جمعا $۲^i * O(n/۲^i) = O(n)$ در هر طبقه زمان می‌برد و مرتبه زمانی کل $O(nlogn)$ خواهد شد.

حافظه مصرفی ما برابر با حافظه مصرفی برای نصف کردن مجموعه نقاط (مجموعه تعداد همه دسته ها همیشه $O(n)$ است) و حافظه مورد نیاز برای پیدا کردن میانه (که در هر لیست متناب با اندازه ان است پس $O(n)$) پس از $O(n)$ است.

پیدا کردن نزدیک ترین نقطه برای پیدا کردن نزدیک‌ترین نقطه ابتدا خود را به این برگ می‌رسانیم. به این شیوه که در گره اول بررسی می‌کنیم ایا x ما کوچک تر است تا با چپ برویم یا خیر و در گره بعدی y و ... . این کار را آن‌قدر تکرار می‌کنیم تا به برگ برسیم. حال نقطه نماینده این برگ می‌تواند یک گزینه خوب برای نزدیک ترین نقطه باشد. فاصله نقطه مورد نظر ما از این نقطه را در نظر بگیریم. اگر این فاصله برابر با r باشد، این اندازه برای ما معیاری خواهد بود تا ناحیه های همسایه را نیز چک کنیم. به این شکل که اگر فاصله عمودی نقطه x از مکعب آن همسایه کمتر از r بود پس آن مکعب مستطیل یک ناحیه احتمالی خوب است. پس وقتی که به عمق برگ می‌رویم دوباره از آن بالا می‌آییم تا همسایه های مناسب را نیز بررسی کنیم. اما این نکته حائز اهمیت است که تعداد همسایه‌های مجاور محدود است و اگر در یک همسایه نقطه ای با فاصله کمتر یافتیم، نقطه کاندیدا و r را متناسب با آن تغییر می‌دهیم پس هر بار در انتخاب همسایه های مجاور سخت‌گیر تر می‌شویم. همچنین اگر یک بار وارد یک شاخه برادر نشدیم، با تنگ‌تر شدن محدودیت فاصله دیگر این اتفاق نمی‌افتاد. و اگر به سمت هر همسایه برویم متناسب با ارتفاع آن زیردرخت تنها عملیات انجام می‌دهیم.

در حالت تصادفی که فاصله نقاط همگن است بیشتر از همسایه های مجاور معکب اولیه را طی نمی‌کنیم پس پیدا کردن نزدیک ترین نقطه موجود ضریب ثابتی (متناسب با k) از ارتفاع درخت یا همان $log(n)$ است.

اضافه کردن یک نقطه آن‌چه نیاز است این است که ناحیه ای که نقطه ما به آن تعلق دارد را پیدا کنیم. به همان شیوه که ابتدا متناسب با این که x آن از ریشه کوچک تر است چپ یا راست برویم و سپس y و ... . زمانی که ناحیه (گره مکعب) که باید به آن اضافه کنیم را این‌گونه یافتیم متناسب با این که از آن بعد مورد نظر (تابع ارتفاع آن گره) بزرگ تر یا کوچک تر است به راست یا چپ آن گره اضافه می‌شود. چون فقط یک بار همچین عملیاتی انجام میشود درخت را می‌توان فرض کرد متوازن است و این درج در زمان ارتفاع انجام می‌شود (کافیست تا برگ مناسب مورد نظر پایین برویم)

پیدا کردن مجموعه نقاط یک ایده: ابتدا باید مکعب مسطیل را پیدا کنیم. در این درخت هر گره نماینده یک معکب مستطیل k بعدی است. بعد پیدا کردن این گره، کافیست یک پیمایش in-order از همه بچه‌هایش بزنیم!

ایده نهایی: بر روی رئوس پیمایش می‌کنیم، و به طور بازگشتی به هر زیر درختی که راس آن عضوی از مکعب مستطیل است وارد می‌شویم .