Petr Kukrál

Referát 36VGE 2006 / 2007

Průnik rovnoběžných obdélníků

1. Abstrakt

Tato práce si klade za cíl seznámit čtenáře s problematikou hledání průniků rovnoběžných obdélníků s rovnoběžnými obdélníky. Práce je zaměřena hlavně na popis vlastního algoritmu vyhledávání průniků a zhodnocení složitosti vzhledem k použité datové struktuře.

2. Motivace

Průnik obdélníků se používá v různých grafických problémech. Kupříkladu vyvíjíme akční hru a do každé úrovně hry umístíme veliké množství herních entit, které při interakci s jinými entitami vyvolají animaci nebo jinou aktivitu. Je to tedy detekce kolizí, které jsou prováděny průnikem hraničních reprezentací těles, což jsou obdélníky v 3D prostoru.

3. Specifikace problému

Budeme uvažovat, že potřebujeme zjistit průnik všech obdélníků se všemi obdélníky. Tato operace se skládá z dvou částí a to prvotní identifikace dvojic obdélníků, které se protínají a z jejich následného hledání vlastního průniku dvojic obdélníků.

Budeme se zbývat pouze první částí problému a to je identifikací všech dvojic obdélníků, které mají nějaký průnik, přičemž nebudeme hledat konkrétní podobu průniku.

4. Přístup k problému

Abychom mohli problém řešit, musíme napřed zjistit, jaká podmínka je nutná k tomu, abychom mohli říci, že se dva obdélníky protínají. Podle [Prep85] je jediná nutná a postačující podmínka, aby dva obdélníky měli společný minimálně jeden bod. Takže budeme testovat průnik intervalů, na což můžeme vhodně použít datovou strukturu Intervalový strom.

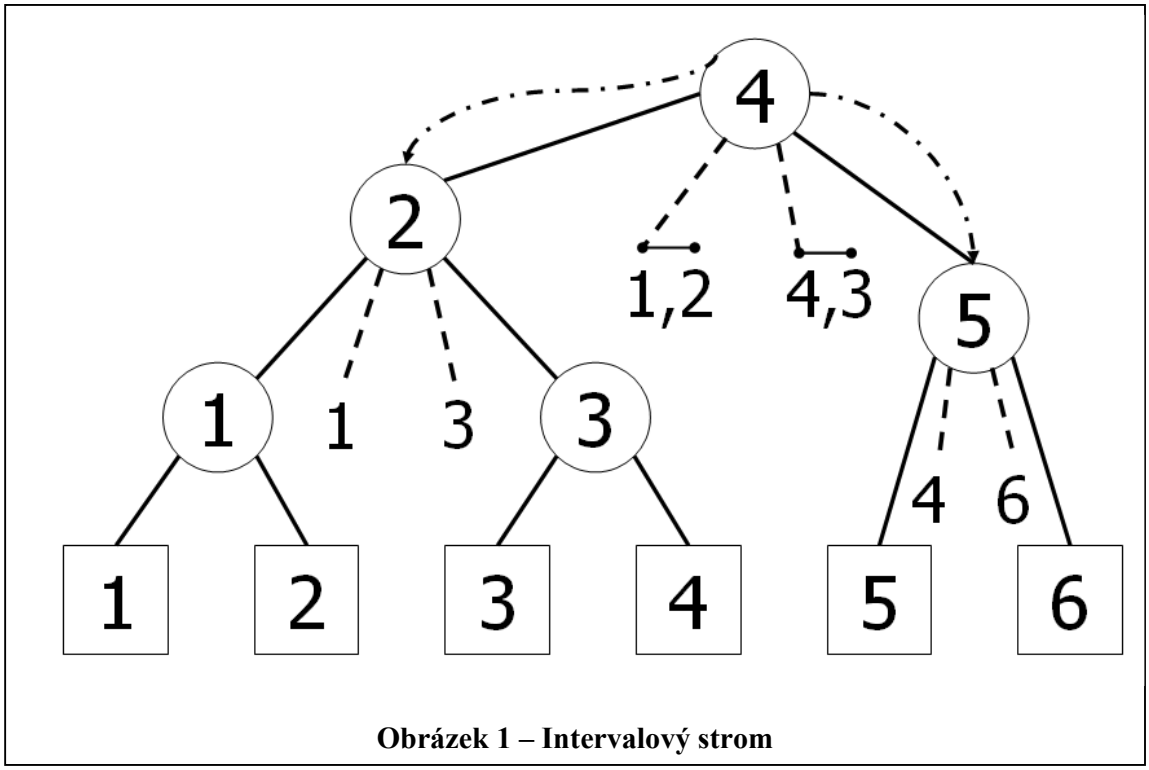
Tím tedy vyřešíme hledání průniku v jedné souřadnici. Potřebujeme zároveň testovat průnik i v druhé souřadnici. Podle [Prep85] je možné vhodně použít současně s intervalovým stromem zametací techniku a dosáhneme tak požadovaného cíle.

5. Popis datové struktury

Na řešení této úlohy je třeba podle [Prep85] modifikovat intervalový strom tak, že do něj přidáme další struktury (viz. Obrázek 1). Přidali do něj super strukturu a substrukturu. Zavedli pojem aktivní obdélník, který označuje, že obdélník (může jich být více) má se zametací přímkou nenulový průnik, jinak je označován jako neaktivní. Uzly, jejichž hodnost patří do intervalu aktivního obdélníku se nazývá aktivní uzel.

Listy stromu (na Obrázku 1 vyznačeny čtverečkem) jsou seřazené souřadnice X hraničních bodů obdélníků. Uzly stromu (na Obrázku 1 jsou to kružnice) reprezentují BVS postavený nad X-ovými souřadnicemi hraničních obdélníků.

Smysl super struktury spočívá v tom, že každý Aktivní uzel má v sobě ještě odkazy LPTR a RPTR. Odkazy uzlu RPTR a LPTR (na Obrázku 1 jsou vyznačeny čerchovanou čárou) ukazují na další aktivní uzly, přičemž RPTR ukazuje na aktivní uzel vpravo a LPTR ukazuje na aktivní uzel vlevo. Dalším přidaným prvkem do intervalového stromu je substruktura, kterou obsahuje každý uzel. Substruktura (na Obrázku 1 vyznačena čárkovanou čárou) a obsahuje ukazatele L(v) – levý seznam a R(v) – pravý seznam. Tyto seznamy obsahují Y souřadnice aktivních obdélníkům, kde levá hranice obdélníku je v levém seznamu



Obrázek 1 – Intervalový strom

a pravá hranice je v pravém seznamu. Levý seznam je seřazen vzestupně a pravý seznam je seřazený sestupně.

6. Postup vyhledávání průniků obdélníků

Zametací přímka je vodorovná přímka, která se pohybuje ve směru X po hraničních bodech obdélníků, seřazených podle X.

Jakmile protne zametací přímka levou hranici obdélníka, nastává první fáze algoritmu. Obdélník označím jako aktivní a začnu hledat fork uzel, což je první uzel jehož hodnost patří do intervalu nového aktivního obdélníku. Cestou k uzlu fork prohledávám substruktury uzlů na průnik s hledaným intervalem. L(v) seznamy uzlu prohledávám pokud cestou z aktuálního uzlu jdu doleva a R(v) pokud jdu cestou doprava. Až najdu fork uzel, nalezl jsem zároveň v tomto uzlu intervaly, které se protínají s aktuálním intervalem. Nový interval vložím do substruktury uzlu. Pokud existují odkazy z tohoto fork uzlu na další aktivní uzly v super struktuře, pak substruktury těchto aktivních uzlů obsahují intervaly obdélníků, které musím také otestovat na průnik.

Jakmile zametací přímka protne pravou hranici obdélníka, nastává druhá fáze algoritmu. Pokud zároveň zametací přímka protne pravý i levý hraniční bod obdélníků, zpracuji napřed levý bod a pak pravý bod. Protnutí pravého hraničního bodu znamená pouze to, že daný interval odstraníme ze substruktury fork uzlu. Pokud se substruktura fork uzlu vyprázdní, uzel se stane neaktivní a odstraníme jeho odkaz ze super struktury.

7.Algoritmus

Mějme množinu obdélníku, jejichž hranice jsou rovnoběžné s osami x,y souřadného systému. Obdélník je definovaný hraničními body [x1,y1] – levý dolní bod a [x2,y2] – pravý horní bod.

1. Seřaď vzestupně X-ové souřadnice levých a pravých hraničních bodů obdélníků
2. Vytvoř nad X souřadnicemi hraničních bodů modifikovaný intervalový strom (IS)
3. Nastav zametací přímku na první X souřadnici hranice
4. Zametací přímka protnula x1 souřadnici zatím neaktivního obdélníku, budeme tedy vkládat interval <y1, y2> neaktivního obdélníku do fork uzlu
    - Při hledání fork uzlu v IS prohledávej substruktury na průnik s <y1,y2>
        - Pokud jdeš cestou vlevo z uzlu U, prohledej L(v) uzlu U na průnik
        - Pokud jdeš cestou vpravo z uzlu U, prohledávej R(v) uzlu U na průnik
    - Nalezen fork uzel F
        - Vlož do substruktury uzlu F interval <y1,y2>
        - Všechny intervaly v uzlu F mají průnik s intervalem <y1,y2>
        - Uzel F označ Aktivní a přidej jeho odkaz do super struktury
    - Z fork uzlu F prohledej všechny substruktury aktivních uzlů pod uzlem F pomocí odkazů v super struktuře
5. Zametací přímka protnula x2 souřadnici již aktivního obdélníku, budeme odstraňovat interval <y1,y2> aktivního obdélníku z fork uzlu
    - Najdi v super struktuře fork uzel intervalu
    - Odstraň ze substruktury interval <y1,y2>
    - Pokud je substruktura fork uzlu F prázdná, označ uzel F Neaktivní a odstraň jej ze super struktury
6. Pokud neprotnula zametací přímka poslední X souřadnici hraničního bodu, posuň zametací přímku na další X souřadnici hraničního bodu a vrať se na krok 3

8. Výsledky

Získání odpovědi ve formě seznamu všech dvojic protnutých obdélníků má výpočetní složitost O(N log N) s paměťovou složitostí S(N). Příprava pro zkonstruování stromu trvá O(N log N) včetně setřídění úseček. Intervalový strom má navíc vlastnost, že vložení a smazání intervalu ze stromu nám trvá čas O(log N). Další výhoda tohoto algoritmu je, že při vkládání intervalu neprovádíme žádný průchod seznamem navíc, neboť interval vložíme tehdy, když narazíme na jeho místo. Porovnání časů na testovacích datech jsem neměl k dispozici, takže jej nemohu doložit.

9. Závěr

Díky vhodné kombinaci známých přístupů, jako jsou Zametací technika a Intervalový strom jsme dokázali vyřešit složitý problém hledání průniku obdélníků ve velice dobrém výpočetním čase. Naivní algoritmus by tento úkol spočítal v čase $O(n^2)$. Tento algoritmus je již několik let otestován a je tedy možné jej použít a spolehnout se na něj v praxi.

10. Literatura

[Prep85] PREPARATA F. P., AND SHAMOS M. I. 1985. Computational Geometry. Springer-Verlag, Berlin