Horváth Mátyás Gergő

BT6Y3A

Asztali szerepjáték támogatása saját nyelv segítségével

Optimalizáció

|  |  |  |
| --- | --- | --- |
| **Eredeti** | **Optimalizált** |  |
| SET(<reg>, <value>)  SET(<var>, <reg>) | SET(<var>, <value>) |  |
| SET(<reg1>, <value1>)  SET(<reg2>, <value2>)  [OP](<reg1>, <reg2>, <var>) | [OP](<value1>, <value2>, <var>) |  |
| [OP](..., <reg>)  SET(<var>, <reg>) | [OP](..., <var>) | [OP] = ADD, SUB, DIV, MUL, OF… |
| x: JUMP(y)  ...  y: JUMP(z) | x: JUMP(z)  ...  y: JUMP(z) |  |
| x: JF(y)  ...  y: JUMP(z) | x: JF(z)  ...  y: JUMP(z) |  |
| SET(<reg>, <obj>)  OF(<prop>, <reg>, <var>) | OF(<prop>, <obj>, <var>) |  |
|  |  |  |

Regiszterek kiküszöbölése

A fordító szinte minden számításhoz regisztereket használ a részeredmények tárolásához, olyan helyeken is, ahol közvetlenül értékeket vagy változókat is használhatnánk. Egy viszonylag egyszerű módszerrel ellenőrizhetjük, hogy mely utasítások esetén cserélhetjük ki a regisztereket konkrét változókra/értékekre.

Fogalmak

A következőkben az alábbi fogalmakat, kifejezéseket használjuk:

* Azt mondjuk, hogy egy utasítás **felhasznál** egy regisztert, ha valamely bemenő paraméterében az a regiszter szerepel.
* Egy utasítás **beállít** egy regisztert, ha eredményét e regiszterbe teszi.
* Egy utasítás **elfogyaszt** egy regisztert, ha egyszerre felhasználja és beállítja azt.
* **Egyszerű változó** egy névvel azonosított változó/objektum (pl. x, sword, actor)
* **Egyszerű érték**nek hívunk egy egyszerű változót vagy bármilyen konstanst (pl. true, 23, $TEXT).

Egyszerű értékre beállított regiszterek kiküszöbölése

SolveSimpleValueRegs

Ha egy SET utasítás beállít egy regisztert egy egyszerű értékre, akkor minden ezt követő utasításban minden paraméter mely e regiszterre hivatkozik kicserélhető az egyszerű értékre addig, amíg újra be nem állítjuk a regisztert. Példa:

|  |  |  |
| --- | --- | --- |
| **Eredeti** | **1. lépés** | **2. lépés** |
| SET(\_0, 10)  SET(\_1, 8)  ADD(\_0, \_1, \_0)  SET(x, \_0) | SET(\_1, 8)  ADD(10, \_1, \_0)  SET(x, \_0) | ADD(10, 8, \_0)  SET(x, \_0) |

A példában először a SET(\_0, 10) utasítást dolgozzuk fel. Kicseréljük az összes \_0 paramétert a 10 értékre egészen addig, amíg \_0-t újra be nem állítja az ADD utasítás. A második lépés a SET(\_1, 8) hasonló kiküszöbölése.

Célregiszter cseréje célváltozóra

SolveRegisterBuffers

Ha egy utasítás az eredményét egy regiszterbe teszi, de ezt a regisztert következő beállításáig csak egyszerű változók értékének beállítására használjuk, akkor az utasítás eredményét teheti közvetlenül az első ilyen változóba (ennek beállító utasítását elhagyhatjuk), minden további helyen pedig kicserélhetjük a regisztert e változóra.

|  |  |
| --- | --- |
| **Eredeti** | **1. lépés** |
| ADD(10, 8, \_0)  SET(x, \_0)  SET(y, \_0) | ADD(10, 8, x)  SET(y, x) |

Többszörös ugrások kiküszöbölése

SolveDoubleJumps

Ha egy ugró utasítás egy JUMP típusú ugró utasításra mutat, akkor az előbbi ugró utasítás célpozícióját a második ugró utasítás célpozíciójára állíthatjuk. Az eljárás – megvalósítástól függően – esetleg többször lefuttatandó, mert egy módosítás következtében lehet, hogy korábbi utasításokat ismét frissítenünk kell. Példa:

|  |  |  |  |
| --- | --- | --- | --- |
| **Eredeti** | **1. lépés** | **2. lépés** | **3. lépés** |
| 0. JF(i, 3)  1. SET(x, 10)  2. SET(y, 20)  3. JUMP(5)  4. SET(z, 30)  5. JUMP(6) | 0. JF(i, **5**)  1. SET(x, 10)  2. SET(y, 20)  3. JUMP(5)  4. SET(z, 30)  5. JUMP(6) | 0. JF(i, 3)  1. SET(x, 10)  2. SET(y, 20)  3. JUMP(**6**)  4. SET(z, 30)  5. JUMP(6) | 0. JF(i, **6**)  1. SET(x, 10)  2. SET(y, 20)  3. JUMP(6)  4. SET(z, 30)  5. JUMP(6) |

Egymás utáni ugrások kiküszöbölése

SolveAdjacentJumps

Ha egymást közvetlenül követő sorokban ugyanarra a pozícióra történő ugró utasítás található, melyek közül legalább egy JUMP típusú, akkor ezek közül elég egyetlen JUMP típusút megtartani. Az alábbi példában az ugró utasítás pozíciója megváltozik a sortörlés következtében.

|  |  |
| --- | --- |
| **Eredeti** | **1. lépés** |
| 0. JF(i, 3)  1. JUMP(3)  2. SET(x, 10) | 0. JUMP(2)  1. SET(x, 10) |

Hasonló a helyzet, ha ugyanazon feltételre hivatkozó JF típusú ugró utasítások vannak egymást közvetlenül követő sorokban. Ebben az esetben elég egyet megtartani közülük.

|  |  |
| --- | --- |
| **Eredeti** | **1. lépés** |
| 0. JF(i, 3)  1. JF(i, 3)  2. SET(x, 10) | 0. JF(i, 2)  1. SET(x, 10) |

Dead code eltávolítás

SolveDeadCode

asd