[**图博弈与字符hash**](http://www.cppblog.com/notonlysuccess/archive/2009/03/01/75212.html)

再来一道简单点的。。。这道用邻接表+hash就OK了  
<http://acm.hdu.edu.cn/showproblem.php?pid=1524>   
  
神奇的图博弈   
<http://acm.hdu.edu.cn/showproblem.php?pid=1809>  
根据递归求出没副图的sg值，然后异或  
但是如果暴力递归的话会做很多重复的搜索  
所以hash一下用记忆化搜索  
要整个地图的状态来hash  
我搞了半天搞不定  
后来查到神奇的字符hash方法  
用了一下果然很灵  
不过现在还是不太清楚为什么这样可以hash

// RS Hash Function  
unsigned int RSHash(char\* str)  
{  
    unsigned int b = 378551 ;  
    unsigned int a = 63689 ;  
    unsigned int hash = 0 ;  
    while (\*str)  
    {  
        hash = hash \* a + (\*str ++ );  
        a \*= b;  
    }  
    return (hash & 0x7FFFFFFF );  
}  
  
// JS Hash Function  
unsigned int JSHash(char\* str)  
{  
    unsigned int hash = 1315423911 ;  
    while (\*str)  
    {  
        hash ^= ((hash << 5 ) + (\*str ++ ) + (hash >> 2 ));  
    }  
    return (hash & 0x7FFFFFFF );  
}  
  
// P. J. Weinberger Hash Function  
unsigned int PJWHash(char\* str)  
{  
    unsigned int BitsInUnignedInt = (unsigned int )( sizeof (unsigned int)\*8 );  
    unsigned int ThreeQuarters = (unsigned int )((BitsInUnignedInt\*3 ) / 4 );  
    unsigned int OneEighth = (unsigned int )(BitsInUnignedInt / 8 );  
    unsigned int HighBits = (unsigned int )( 0xFFFFFFFF ) << (BitsInUnignedInt - OneEighth);  
    unsigned int hash = 0 ;  
    unsigned int test = 0 ;  
    while (\*str)  
    {  
        hash = (hash << OneEighth) + (\*str ++ );  
        if ((test = hash & HighBits) != 0 ) {  
            hash = ((hash ^ (test >> ThreeQuarters)) & ( ~ HighBits));  
        }  
    }  
    return (hash & 0x7FFFFFFF );  
}  
  
// ELF Hash Function  
unsigned int ELFHash(char\* str)  
{  
    unsigned int hash = 0 ;  
    unsigned int x = 0 ;    while (\*str)  
    {  
        hash = (hash << 4 ) + (\*str ++ );  
        if ((x = hash & 0xF0000000L ) != 0 ) {  
            hash ^= (x >> 24 );  
            hash &= ~ x;  
        }  
    }  
    return (hash & 0x7FFFFFFF );  
}  
  
// BKDR Hash Function  
unsigned int BKDRHash(char\* str)  
{  
    unsigned int seed = 131 ; // 31 131 1313 13131 131313 etc..  
    unsigned int hash = 0 ;  
    while (\*str)  
    {  
        hash = hash\*seed + (\*str ++ );  
    }  
    return (hash & 0x7FFFFFFF );  
}  
  
// SDBM Hash Function  
unsigned int SDBMHash(char\* str)  
{  
    unsigned int hash = 0 ;  
    while (\*str)  
    {  
        hash = (\*str ++ ) + (hash << 6 ) + (hash << 16 ) - hash;  
    }  
    return (hash & 0x7FFFFFFF );  
}  
  
// DJB Hash Function  
unsigned int DJBHash(char\* str)  
{  
    unsigned int hash = 5381 ;  
    while (\*str)  
    {  
        hash += (hash << 5 ) + (\*str ++ );  
    }  
    return (hash & 0x7FFFFFFF );  
}  
  
// AP Hash Function  
unsigned int APHash(char\* str)  
{  
    unsigned int hash = 0 ;  
    int i;  
    for (i = 0 ;\*str; i ++ )  
    {  
        if ((i & 1 ) == 0 )    {  
            hash ^= ((hash << 7 ) ^ (\*str ++ ) ^ (hash >> 3 ));  
        }    else {  
            hash ^= ( ~ ((hash << 11 ) ^ (\*str ++ ) ^ (hash >> 5 )));  
        }  
    }  
    return (hash & 0x7FFFFFFF );  
}