Challenge January 2025

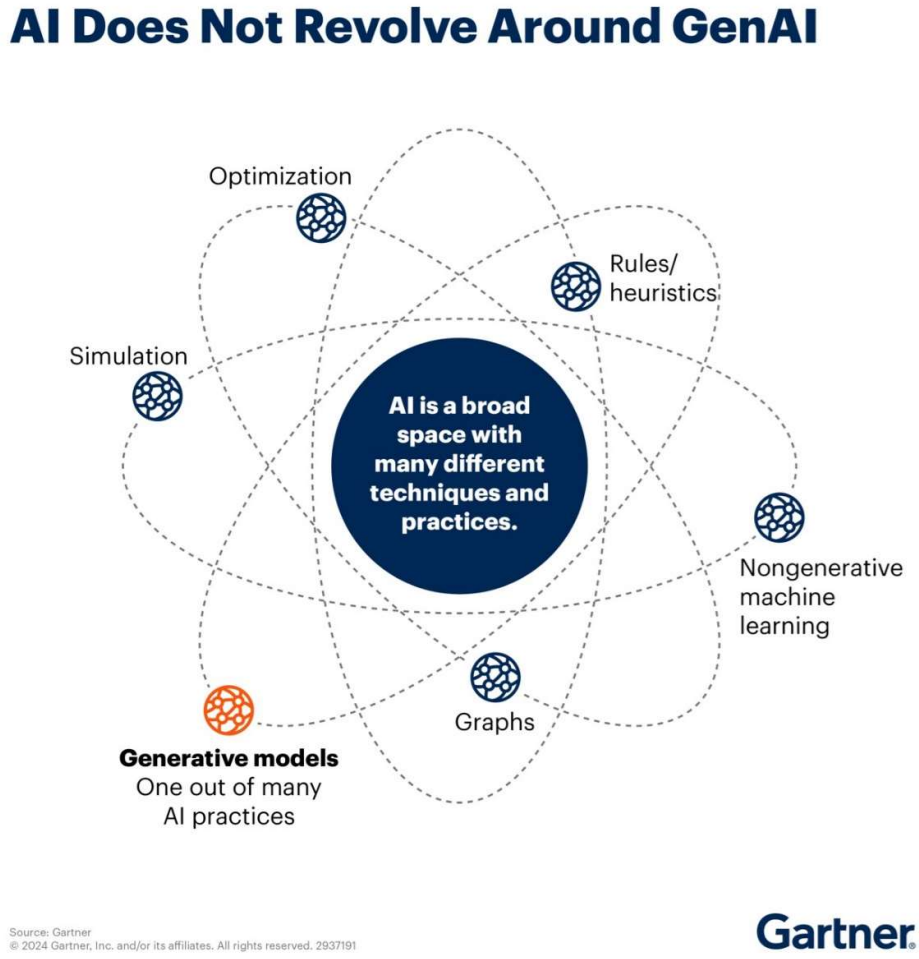
Christmas Word Search

A solution with OPL CPLEX by Alex Fleischer
afleischer@fr.ibm.com

CPLEX is available within IBM AI platform watsonx.AI
And OPL is a high level modeling language for CPLEX

This search for words looks like a combinatorial problem so within CPLEX I chose to use the Constraint Programming engine CPOptimizer.

I have not used any generative AI here since I want to remind everybody that AI is wider than genAI as Gartner wrote.



In OPL I wrote

```
using CP;

string s="\
MMMSXXMASM\
MSAMXMSMSA\
AMXSXMAAMM\
MSAMASMSMX\
XMASAMXAMM\
XXAMMXXAMA\
SMSMSASXSS\
SAXAMASAAA\
MAMMMXMMMM\
MXMXAXMASX\
";
```

```
string s2="XMAS";

// length of string s
int slength;
// length of string s2 ("XMAS")
int s2length;
// dimension of the grid
int dim;

execute
{
  slength=s.length;
  dim=Math.sqrt(slength);
}

assert dim*dim==slength;

range r=1..dim;

int grid[r][r];

// Translate to a 2D array of ASCII codes
execute turn_string_into_grid
{
  var index=0;
  for(var i in r) for(var j in r)
  {
    grid[i][j]=s.charCodeAt(index);
    index++;
  }

  s2length=s2.length;


}

assert s2length<=dim;

range r2=1..s2length;
```

```
int grid2[r2];

// Translate into a 1D array of ascii codes
execute
{
  for(var i in r2) grid2[i]=s2.charCodeAt(i-1);
}

// Decision variables

// Where do we pick letters ?
dvar int x[r2] in r;
dvar int y[r2] in r;

// Constraints

subject to
{
      // Read the right word
      forall(i in r2) grid[x[i],y[i]]==grid2[i];

      // Same line or column
      ((and(i in r2) x[i]==x[1]) &&
      ((and(j in r2:j!=1)y[j-1]+1==y[j])||(and(j in r2:j!=1)y[j-1]-1==y[j])))
      ||
      ((and(i in r2) y[i]==y[1]) &&
      ((and(j in r2:j!=1)x[j-1]+1==x[j])||(and(j in r2:j!=1)x[j-1]-1==x[j])))

     || // Same diagonal
      ((and(i in r2) x[i]+y[i]==x[1]+y[1]) &&
      ((and(j in r2:j!=1)x[j-1]+1==x[j])||(and(j in r2:j!=1)x[j-1]-1==x[j])))
       ||
      ((and(i in r2) x[i]-y[i]==x[1]-y[1]) &&
      ((and(j in r2:j!=1)x[j-1]+1==x[j])||(and(j in r2:j!=1)x[j-1]-1==x[j])))
 ;
}
```

```
// Display solution
execute
{
  //writeln("x=",x);
  //writeln("y=",y);
  //writeln();
}


// Enumerate all options
main
{
    cp.param.SearchType=24;
    cp.param.workers=1;
    var nb=0;

    thisOplModel.generate();
    cp.startNewSearch();
    while
    (cp.next()) {  thisOplModel.postProcess(); nb++;  }
    writeln("number of solutions = ",nb);
}

/*

which gives

number of solutions = 18

*/
```

And I got

number of solutions = 18