A problem from Technology Review's Puzzle Corner: March 2005

How many integers from 1 to 100 can be made using the digits 2, 0, 0 and 5 exactly once when combined with the operators +, -, *, / and ^? Solutions with fewest operators (parentheses ignored), then with digits in year order are preferred.

A solution by Joel Karnofsky, 2/10/05

First, comments on the published solution for 2004. The expression for 3 can be improved to $2+40^0$. There is a typo in the value for 41, which could properly be 2^0+40. More generally, the instructions are not explicit about whether leading zeroes are allowed in numbers. If they are allowed, no new numbers would be achievable (0xy = 0+xy) but many would have simpler representations, such as 6 = 2+004 and 24 = 0024. In what follows I assume leading zeroes are not allowed.

My approach to the 2005 problem is to write a *Mathematica* program that solves the problem for an arbitrary year. I use that unary minus is never useful as an operator except in an exponent. This follows because only positive results are of interest and fewer operators is preferred to year order. However, for example, for the year 2278, 58 = $(2^{-2}+7)8$ and 58 cannot be achieved otherwise, so negated exponents must be allowed for. I ignore the possibility of negating numbers to be raised to fractional powers. To produce anything useful from the resulting complex numbers would require more digits than we are considering here. Finally, I assume that when a positive number is raised to a fractional power, only the positive root can be used.

Following is code that computes just a list of achievable numbers. More complex code that computes an optimal representation is shown at the end. Starting with the year as a number (*yearNumber*) first compute some related values:

```
year=IntegerDigits[yearNumber];
digits=Length[year];
vars=Array[v,digits];
```

Next, compute all expressions based on the ordered list *vars* using the allowed operators in all parenthesizations. *Power* is replaced by *power* in the list of operators for reasons described below. The implicit operation of combining successive digits is implemented by *ten*, with rules to limit when this operation is allowed.

```
opRules=Append[
     {a___,b_,c_,d___}->{a,#[b,c],d}&/@
          {Plus,Subtract,Times,Divide,power,power[#1,-#2]&},
     {a___,b_v|b_ten,c_v,d___}->{a,ten[b,c],d}
];
insertOps[expr_]:=ReplaceList[expr,opRules];
expressions=First/@Union@Nest[Flatten[insertOps/@#,1]&,{vars},digits-1];
```

Finally, substitute all possible permutations of the year's digits into these expressions and select resulting integers in the desired range. For the variety of undefined or undesired cases, simply ignore the non-numerical results and any associated error messages. The restrictions on *power* are simply to save time and memory by not computing high powers that cannot be of interest.

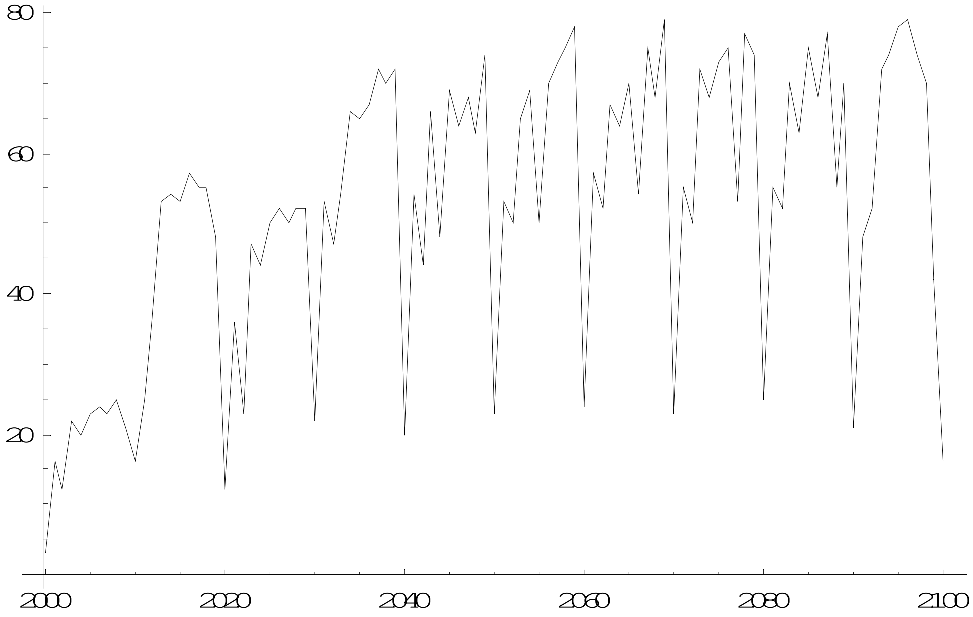
```
Off[Power::"infy",Power::"indet",∞::"indet"];

ten[a_Integer,b_]:=10a+b;

power[a:-1|0|1|_DirectedInfinity|Indeterminate,b_]:=a^b;
power[a_,b_/;Abs[b]<10]:=a^b;
power[_?NumberQ,_]=Indeterminate;

allValues[perm_]:=Cases[expressions/.Thread[vars->perm],n_Integer/;1≤n≤100];
Union@Flatten[allValues/@Permutations[year]]
```

As examples of the use of this code relevant to the published statement of the problem, there will be 25 achievable integers in 2011 (and 2008) and 53 in 2013. The best of the century is 79 in 2069 (or 2096). The best for all four-digit years is 98 in 1389, 2347 or 2379 (or any reordering of these digits). Here is a plot of the number of achievable integers by year for this century:



The code below computes not only achievable numbers for a year but also gives optimal representations. When there is more than one optimal representation, the code picks one with the fewest total *Subtract* and *Divide* operators; which still may not be unique. For year 2005 it finds the following 23 formulas. (The unnecessary parenthesis in 15 = (20+0)–5 can be taken as a program bug.)

```
  1 = 205^0
  2 = 0^50+2
  3 = 2+50^0
  4 = (20+0)/5
  5 = 20^0×5
  6 = 20^0+5
  7 = 2+0+0+5
 10 = (2+0+0)×5
 15 = (20+0)-5
 19 = 20-5^0
 20 = 20+0^5
 21 = 20+5^0
 25 = 20+0+5
 30 = 50-20
 32 = (2+0)^(0+5)
 40 = 200/5
 48 = 0-2+50
 49 = 50-2^0
 50 = 0^2+50
 51 = 2^0+50
 52 = 0+0+52
 70 = 20+50
100 = (20+0)×5
```

The logic of the following code is essentially the same as that above, with additions to rank all representations of a number. However, to produce properly formatted formulas it depends on *Mathematica*-specific symbols that are predefined as operators with specified syntax but no defined semantics. The resulting expressions are converted to strings and the symbols converted to the expected ones; all to avoid *Mathematica's* sometimes over zealous desire to format expressions its way. I won't attempt to document this code in detail.

```
formulas[yearNumber_Integer]:=Block[{SmallCircle,CirclePlus,Star},

     year=IntegerDigits[yearNumber];
     digits=Length[year];
     vars=Array[v,digits];

     SetAttributes[{CirclePlus,Star,SmallCircle},Flat];

     opConvert=Thread[
          {CirclePlus,CircleMinus,Star,Diamond,CircleDot,Square,SmallCircle}->
          {Plus,Subtract,Times,Divide,power,Minus,fromDigits}
     ];
     charConvert=Thread[
          {" "," ",".", " "," "," "," "," "," "}->
          {"+","-","×","/","^","-",""," = ",""}
     ];
     opRules=Append[
          {a___,b_,c_,d___}->{a,#[b,c],d}&/@
          {CirclePlus,CircleMinus,Star,Diamond,CircleDot,CircleDot[#1,Square[#2]]&},
          {a___,b_v|b_SmallCircle,c_v,d___}->{a,SmallCircle[b,c],d}
     ];
     insertOps[expr_]:=ReplaceList[expr,opRules];
     expressions=First/@Union@Nest[Flatten[insertOps/@#,1]&,{vars},digits-1];
     expressions=Thread[{expressions/.opConvert,expressions}];

     Off[Power::"infy",Power::"indet",∞::"indet"];

     fromDigits[0,__]=Indeterminate;          (* delete to allow leading zeroes *)
     fromDigits[a_Integer,b__]:=FromDigits[{a,b}];

     power[a:-1|0|1|_DirectedInfinity|Indeterminate,b_]:=a^b;
     power[a_,b_/;Abs[b]<10]:=a^b;
     power[_?NumberQ,_]=Indeterminate;

     allFormulas[perm_]:=Cases[
          expressions/.Thread[vars->perm],{n_Integer/;1≤n≤100,expr_}:>{
               n,
               (Plus@@Cases[expr,SmallCircle[v__]:>1-Length[{v}],{0,Infinity}])+
                    Count[expr,_Square,{0,Infinity}],
               perm=!=year,
               Count[expr,_CircleMinus|_Diamond,{0,Infinity}],
               StringReplace[ToString[n  expr],charConvert]
          }
     ];
     Split[Sort[Flatten[allFormulas/@Permutations[year],1]],
        First@#1==First@#2&
     ][[All,1,5]]
]
```