Quarter polygon with least area covering a quarter circle with radius n

I) Examples for $1 \le n \le 17$

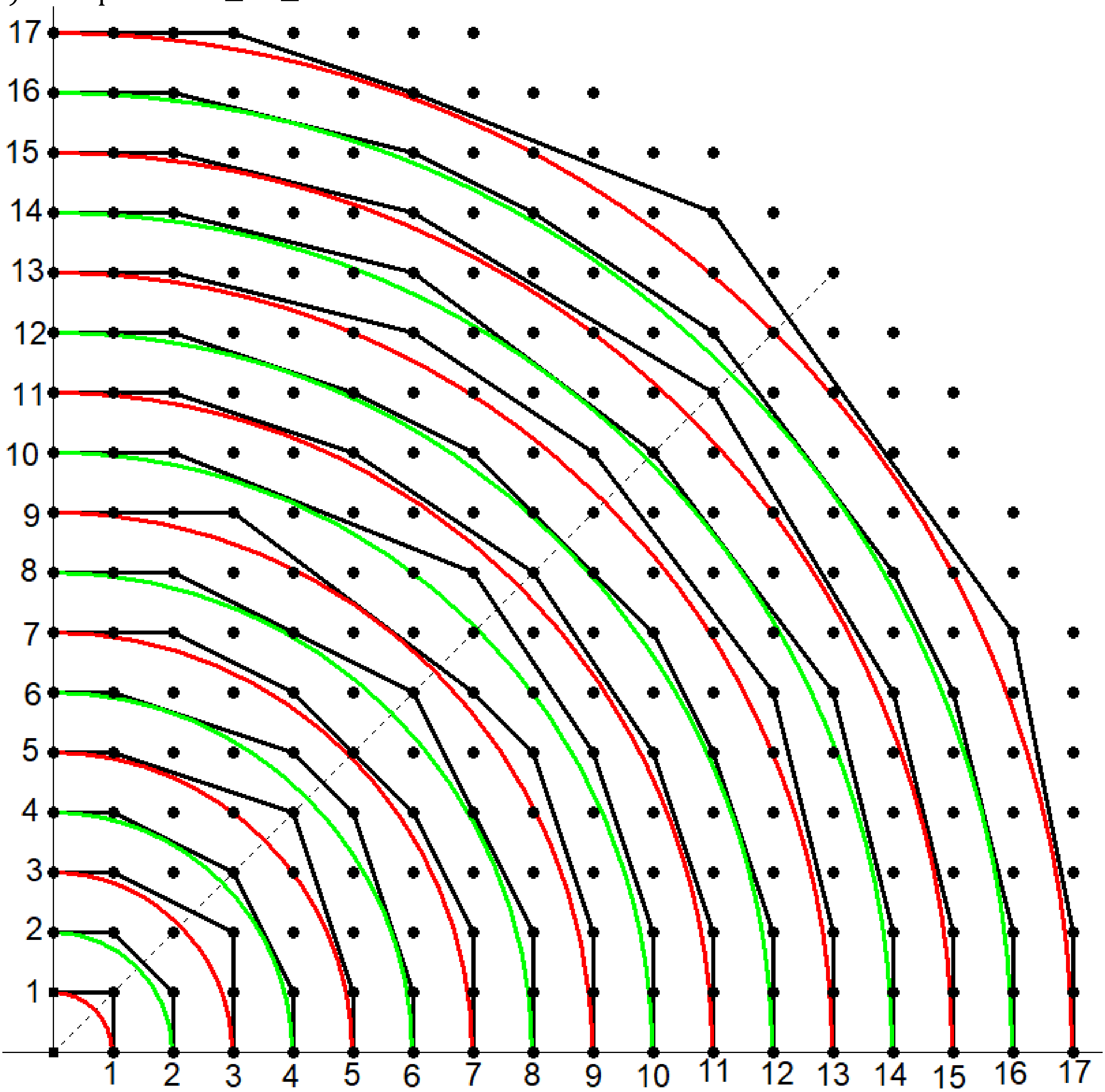

Fig. 1

II) Quality of a path

It is assumed that the distance of any vertex from the circle is <1, see V), annotations. Fig. 2a repeats the example for n=5 with the doubled sector areas such that $a(5) = 42$ can be verified. Fig. 2b describes, for the same example, details of the algorithm.

The path begins at (0,5) and ends at (5,0). Then 9 further points may be located on the polygon: (1,5); (2,5); (3,5); (3,4); (4,4); (4,3); (5,3); (5,2); (5,1). Each path leading to a point $P(x, y)$ has a quality described by three parameters (A, dx, dy) where A is the doubled area enclosed by the partial polygon and $(x - dx, y + dy)$ the predecessor of P.

Restrictions: $dy > 0$ and $dx > 0$. Exceptions: $dy = 0$ for $y = n$ and $dx \geq 0$ for x$= n$.
Generally several paths lead to P. Therefore, P is associated with a set of qualities.
There are two aspects of a quality: The smaller the area and the flatter the edge the better is the quality. If a quality is worse than another one considering both aspects, it can be ignored. Generally a set can be reduced by ignoring bad qualities.
In fig. 2b, none of the two qualities (34,2,3) and (33,1,2) can be ignored because 34>33, but $\frac{3}{2} < \frac{2}{1}$.

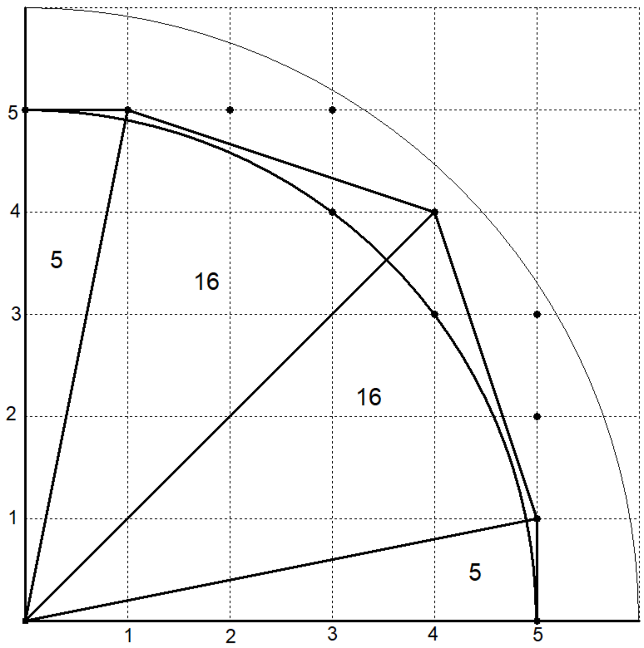

Fig. 2a

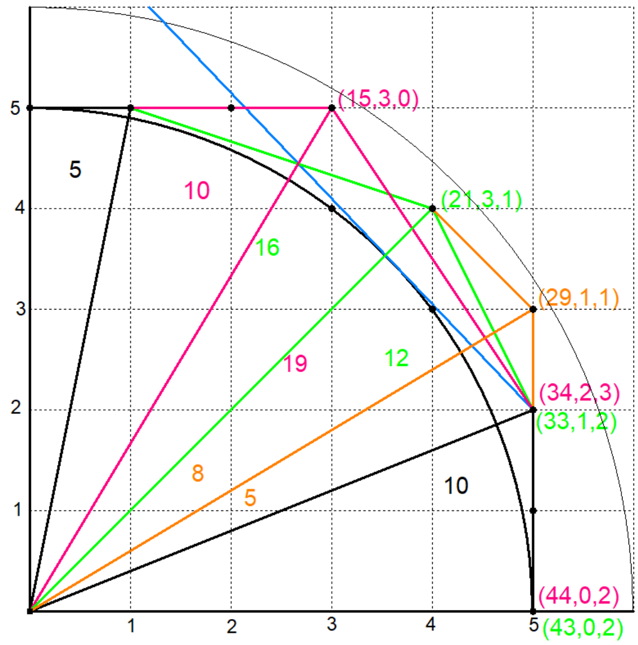

Fig. 2b

Let us consider polygons with point (5,2). Regarding convexity, predecessors must be located on the right side of the tangent, namely (3,5), (4,4) and (5,3). The quality of the amber path in (5,2) is (34,1,0) and can be ignored by comparison with (34,2,3).

Considering convexity, a flatter edge generally allows a greater choice of succeeding edges: In fig. 3, the blue edge may be a successor of the red, but not of the green one. Reduction rule:
Let (A_1, dx_1, dy_1) and (A_2, dx_2, dy_2) two qualities in the same set and $A_1 \leq A_2$ and $dy_1 \cdot dx_2 \leq dy_2 \cdot dx_1$. Then the second quality can be ignored.
Note: $dy_1 \cdot dx_2 \leq dy_2 \cdot dx_1$ instead of $\frac{dy_1}{dx_1} \leq \frac{dy_2}{dx_2}$ avoids a possible division by zero.

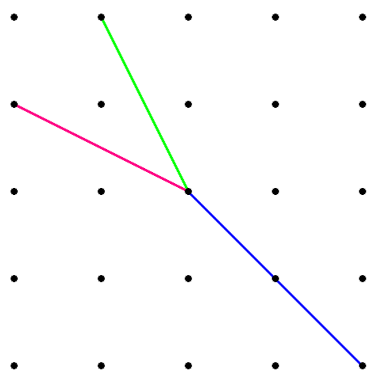

Fig. 3

The quality sets for n=5 after reduction:
(1, 5): {(5, 1 , 0)}   (2, 5): {(10, 2 , 0)}   (3, 5): {(15, 3 , 0)}
(3, 4): {(17, 1, 1)}   (4, 4): {(21, 3, 1)}
(4, 3): {(26, 1, 2) ; (25, 0, 1)}
(5, 3): {(29, 3, 2)}   (5, 2): {(34, 2, 3) ; (33, 1, 2)}
(5, 1): {(37, 1, 3)}   (5, 0): {(42, 0, 1)}
The path with the least area 42 can be reconstructed backwards:
(5,0)→(5-0,0+1)=(5,1) →(5-1,1+3)=(4,4) →(4-3,4+1)=(1,5) →(0,5)

III) Formulas

a) Restriction on a predecessor(x_0, y_0) of a point (x, y) with $y > 0$:

$$y_0 \geq y + m \cdot (x - x_0)$$

where m is the negative gradient of the tangent (such that m is nonnegative):

$$m = \frac{n^2 - y^2}{2 \cdot x \cdot y} \text{ for } x = n \text{ and } m = \frac{x \cdot y - n \cdot \sqrt{x^2 + y^2 - n^2}}{n^2 - x^2} \text{ for } x < n$$

b) Further restriction on (x_0, y_0) with a given quality (A_0, dx_0, dy_0):

$$dy_0 \cdot dx \leq dy \cdot dx_0 \text{ with } dx = (x - x_0) \text{ and } dy = (y_0 - y)$$

This leads to a quality for (x, y): $(A_0 + \Delta A, dx, dy)$ where $\Delta A = x \cdot y_0 - x_0 \cdot y$ is the doubled area of the triangle (x_0, y_0) $(0,0)$ (x, y).

IV) The algorithm

a) Select $h > 0$ such that the maximum distance of a polygon point from the circle is presumed to be $< h$, for example $h = 1$.
b) For the points (x, y) with $n^2 \leq x^2 + y^2 < (n + h)^2$, set the order “row by row (top down) and in a row from left to right”.
c) Find all predecessors (x_0, y_0) of (x, y) regarding III).
d) Find a quality (A_0, dx_0, dy_0) such that $A = A_0 + \Delta A$ is the minimum, see IIIb).
e) Add (A, dx, dy) to the set of qualities for (x, y) and reduce the set.
f) The quality set of the last point (n,0) is $\{(a(n), 0, dy)\}$, see II), last paragraph.

Annotations:

1) The evaluation for $1 \leq n \leq 1000$ yields that the maximum distance of a polygon point from the circle is $\sqrt{317} - 17 \approx 0.8$ for n=17, see fig. 1. Varying h from 0.81 to 1.5, this result is stable. As computing time increases with h, h=1 is a good choice.

2) One could presume:
A quality (A_2, dx_2, dy_2) is generally worse than (A_1, dx_1, dy_1) if $A_1 < A_2$. With this modification of the reduction rule, no error occurs for n<136, but for n=136, the result is 29098 instead of 29097.

V) Appendix: VB-code of the algorithm (Excel-macro)

Note: A quality set is encoded by a string with 5 bytes per quality: 3 bytes for the area A and one byte each for dx and dy, see subroutines quality, code and decode. Thus the program can be (theoretically) used up to $n = 5795$.

```
Const nmax = 40, h = 1
Dim qualiset$(nmax, nmax)

Sub polygons_around_a_circle()
Dim x1(nmax), x2(nmax) ' x1(y) to x2(y): range for points(x,y)
prot$ = ""
For n = 1 To nmax
  Erase qualiset$()
  For y = n To 0 Step -1
    If y = 0 Then x2(y) = n Else x2(y) = Int(Sqr((n + h) ^ 2 - y ^ 2))
    If y = n Then
      x1(y) = 1
      For x = 1 To x2(y)
        qualiset$(x, y) = code(x * y) + Chr(x) + Chr(0)
      Next
    Else
      x1(y) = -Int(-Sqr(n ^ 2 - y ^ 2))
      For x = x1(y) To x2(y)
        If y > 0 Then
          If x = n Then
            m = (n ^ 2 - y ^ 2) / (2 * x * y)
          Else
            m = (x * y - n * Sqr(x ^ 2 + y ^ 2 - n ^ 2)) / (n^2 - x^2)
          End If
        End If
        y0 = y
        Do
          y0 = y0 + 1
          If y = 0 Then xa = n Else xa = -Int((y0 - y) / m - x)
          For x0 = xa To x2(y0)
            qualiset$(x, y) = update(n, x0, y0, x, y)
          Next
        Loop Until y0 = n
      Next
    End If
  Next
  a_n = decode(qualiset$(n, 0))
  Cells(n, 1).Value = a_n
  prot$ = prot$ + Str(a_n) + ","
Next
Cells(1, 2).Value = prot$
End Sub

Function update(n, x0, y0, x, y)
  p1$ = qualiset$(x0, y0): p2$ = qualiset$(x, y)
  dx0 = x - x0: dy0 = y0 - y
  dar = x * y0 - y * x0
  armi = 2 * (n + 1) ^ 2
  For j = 1 To Len(p1$) Step 5
    pa = Mid(p1$, j, 5)
    Call quality(pa, ar, dx1, dy1)
    If dy1 * dx0 <= dy0 * dx1 Then
      If ar < armi Then armi = ar
    End If
```

```
    ar = armi + dar

    hk$ = code((ar)) + Chr(dx0) + Chr(dy0)
    If p2$ = "" Then
      p2$ = hk$
    Else
      For k = 1 To Len(p2$) Step 5
        pb = Left(p2$, 5)
        Call reduce(hk$, ar, dx0, dy0, p2$, pb)
        If hk$ = "" Then Exit For
      Next

      If hk$ > "" Then
        s = InStr(p2$, hk$): newquality = (s Mod 5 <> 1)
        If newquality Then p2$ = hk$ + p2$
      End If
    End If
    update = p2$
  Next
End Function

Sub reduce(hk$, ar, dx0, dy0, p2$, pb)
  Call quality(pb, ara, dx2, dy2)
  dif = dy0 * dx2 - dy2 * dx0
  If ar < ara Then
    better = (dif <= 0)
  ElseIf dif >= 0 Then
    hk$ = ""
  Else
    better = (ar = ara)
  End If
  If better Then p2$ = Mid(p2$, 6)
End Sub

Sub quality(p, ar, dx, dy)
  ar = decode(Left(p, 3))
  dx = Asc(Mid(p, 4)): dy = Asc(Mid(p, 5))
End Sub

Function code(q)
  For k = 1 To 3
    p = q Mod 256: w$ = Chr(p) + w$: q = q \ 256
  Next
  code = w$
End Function

Function decode(w$)
  u$ = w$: p = 0
  For k = 1 To 3
    a = Asc(Mid(u$, k)): p = p * 256 + a
  Next
  decode = p
End Function
```