placed on the borderline side-by-side (though eventually one of these pairs of black and grey nodes does not count as a contact, as is made clear below).

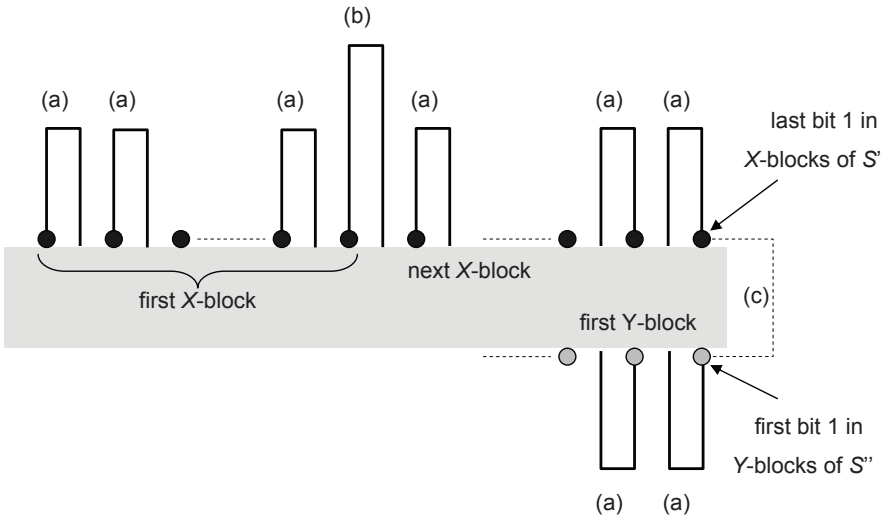

Fig. 6.20. Black nodes represent all bits 1 of X-blocks of string S'. Each of the paths starting above a black node on the borderline consists of an odd number of nodes and ends with a node on the borderline. Bold lines **(a)** indicate blocks of all zeroes that separate consecutive bits 1 within X-blocks; broken lines **(b)** indicate blocks consisting of all zeroes between an X-block and next Y-block together with the complete next Y-block; "turn around" **(c)** consists of an even number of nodes between black and grey dot. Below borderline the same picture results with Y-blocks of string S'' (bits 1 of Y-blocks within S'' are represented by grey dots).

As Fig. 6.19 for the particular example shows, and Fig. 6.20 makes clear for the general case, we manage to create a fold with a number of contacts of bits 1 that is at least $\min\{1/2\,x(S), 1/2\,y(S)\} - 1$. Note where the '-1' comes from. It might be the case that the last 1 in the last X-block of S' is immediately followed by the first 1 in the first Y-block of S'' (for example, think of strings consisting only of bits 1). In that case, this single pair of immediately consecutive bits 1 does not count as a contact. Now we put this number in relation to the maximum possible number max of folds as follows:

$$\begin{aligned} &\max \leq 2\min\{x(S), y(S)\} + 2 \\ &\min\{x(S), y(S)\} \geq \frac{1}{2}(\max -2) \\ &\text{folds} \geq \frac{1}{2}\min\{x(S), y(S)\} - 1 \geq \frac{1}{2}\frac{1}{2}(\max -2) - 1 = \frac{1}{4}\max -\frac{3}{2} \end{aligned} \tag{6.16}$$

This looks almost as expected for a 4-approximation algorithm, if we ignore the constant additive term $-1\frac{1}{2}$. Indeed, it is an example of a 4-approximation