Volume 12 Number 1 1984 Nucleic Acids Research

PMAP, PMAPS: DNA physical map constructing programs

Gábor Polner, László Dorgai and László Orosz

Department of Genetics, Attila József University, 6726 Szeged, Középfasor 52, Hungary

Received 27 July 1983

ABSTRACT

Computer programs are described, which facilitate the construction of the restriction site (physical) map of DNA molecules. By knowing the length of each fragment and its degree of error in the single and the double restriction enzyme digestions, the programs give all the possibilities for the physical map. This method is applicable to linear DNA molecules. Several examples are presented which indicate the high efficiency of the programs in constructing restriction site maps for the 62 Kb chromosome of bacteriophage 16-3. We have constructed complex maps (i.e. EcoRI map with 16 and EcoRV with 11 fragments).

INTRODUCTION

In the last decade physical maps have developed an important role in molecular biology. There are techniques to determine the lenght of the fragments of a DNA molecule digested with a restriction enzyme (1.). But when the number of fragments in a digestion is numerous (about 10-15) it is a very tiring and takes excessive amounts of time to construct the physical map.

There are two earlier papers about physical map construction utilizing the computer (2.), (3.). Stefik's program (2.) constructs the physical map by permutating the fragments of the double digest and the recognition sites. This program uses a rule system to evaluate the possible permutations. Pearson's program is described as a faster one (3.). This program permutates the fragments of the single digestions, and compares the actual and hypothetical double digest's fragments.

If we know the fragment order of the single and double digests it is possible to optimize the fragments' length using the computer program of Schroeder and Blattner (4.).

© IRL Press Limited, Oxford, England.

In the physical map construction our programs aid in the following way: knowing the fragment lengths and their errors in two single restriction digestions of the DNA molecule, and knowing the fragment lengths and their errors in the double digestion, the programs construct all the possibilities for the physical map of the two singles as well as of the double digestion. It must be noted, that this procedure may be applicable only to a linear molecule. If the DNA molecule has a circular form, first it must be converted into linear form. This can be achived by dissociation at the cohesive ends, if there are any cohesive ends, or by digestion with a restriction enzyme which has only one recognition sequence along the DNA molecule.

THE ALGORITHM OF THE PROGRAMS

The algorithm builts up from left to right the physical maps of the two singles and of the double digest simultanously, fragment by fragment. The basic algorithm uses the following rules (the two single digestion fragment sets are designated with α and β):

B.1. The first fragment of the double digestion's physical map and that of the adequate single digestion's map must be equal.

B.2.a. Let us suppose that the right hand restriction site of the last double digestion fragment placed belongs to α ! If the next cleavage site of the next double digestion belongs to α, the just placed single and double digestion fragments must be equal.

B.2.b. If the next cleavage site of the double digestion belongs to β (the last belonged to α), the fragment must be equal to the sum of the double digestion fragments fitted in the construction after the last β fragment.

B.3. If there is no continuation possibility listed in B.2. we must change the last placed double digestion and the single digestion fragment for fragments that have not been tried as described in B.2. and try to continue the construction.

B.4. If we have used up all fragments in the construction we have got a physical map. We dizmamtel the map from the right end checking for different variations of the physical map with

the removal of each fragment as in B.3..
B.5. The procedure ends, if we do not have more starting possibility for the map.

To make the algorithm quicker and the output simpler the following tricks are used:
T.1. If an α fragment contains n β restriction sites ($n \geq 2$) there are n-1 β fragments who's sum is equal to the original α fragment of the double digest. The order of these β fragments is not decideable with this double digest and the algorithm would supply them in all permutations. To avoid this redundancy the algorithm considers only one permutation and reveals this permutation in brackets in the final physical maps. Moreover the algorithm saves the last 20 orders and constructs only such maps which are different from the saved ones.
T.2. Let us suppose that the last placed fragment belongs to β and the next α fragment is fixed (we can give this information as input data)! In this case, to avoid the permutations coming from placing many β fragments equal in size to the double digest fragments, we check wether the fixed α fragment may be a continuation of the map in hand. If this condition is not realized, we proceed as described in B.3. and try to continue the construction.
T.3. If the terminal fragments of the maps are not known the algorithm would supply every physical map twice, from left to right and vica verse. Avoiding this redundancy can reduce the output and the running time in half. When we have a variation for the whole physical map, the last fragment of this map can be excluded as a new starting possibility for the next variation of the physical map.
T.4. If we have two different fragments equal in size (i.e. "double band" in gels) then the algorithm would supply identical orders as if they were different versions when these fragments are exchanged with each other. To avoid this redundancy the algorithm keeps only one continuation possibility of the construction for the continuation possibilities with fragments of the same length.
T.5. By giving a parameter it is possible to construct only a beginning section of the physical map instead of the whole. In

this case we have to give the number of double digestion fragments to map physically. This trick makes possible to reduce the size of the problem (i.e. there is a big fragment on the α map's border and to this region there are mapped several β fragments.)

T.6. In practice in the first step we are usually interested in the physical map of only one enzyme and not in the fragment order of the double digestion. The algorithm can produce the maps of the asked enzyme as it is reached, and finishing the procedure the maps for the asked enzyme are listed. (If the same order belongs to several double digest maps it is listed only once.)

T.7. It is very hard to estimate the running time needed by the programs. To solve this problem in every 20-30 CPU seconds the programs write out briefly where the procedure is in the construction of a physical map. (This is only a few lines of information.) If we want to go on with the construction, we put in the last saved result as input data and the programs can continue running.

If we have some information about the physical maps (results of special digests), we can determine them more quickly. The following possibilities illustrate this aspect of the algorithm:

S.1. We can classify the fragments into groups and we can give the order of these groups as input data. This information can be the physical map of one of the enzymes, or only which fragments are the cohesive ends, or the physical map for one enzyme plus restriction data for the second enzyme for those fragments which are situated at either end of the known physical map.

Up to this point we described the features characteristic of the PMAP program.

S.2. The PMAP program cannot take into consideration the results of digests of deletion mutants and unmapped isolated fragments. These results are useable only to select the possible map variations given by the program. The information, which fragments are grouped on the basis of the special digests, (digests of deletion mutants and isolated fragments), can be given as input data. So the algorithm will construct

only such maps where the given fragment group does not contain any stranger fragment.

Up to this point we described the features characteristic of the PMAPS program. (The difference is only S.2..)

THE OPERATION-CONSUME OF THE ALGORITHM

We will count how many times we must check a continuation of the physical map in hand to see whether it is a good continuation.

Let us denote by $r(n)$ the number of all the physical maps in hand, in which n double digestion fragments are placed. So the number of checks will be:

$$C=\sum_{k=o}^{n_{ab}-1}(n_{ab}-k)(n_a+n_b-k)r(k)+r(n_{ab})$$

Where n_a, n_b, n_{ab} denote the number of fragments of the two single and the double digestions. $r(n)$ can be determined by:

$$r(n+1)=\sum_{i=1}^{r(n)} c(i,n) \quad ; \quad r(o)=1$$

Where $c(i,n)$ denotes the number of the possible continuations of the i-th physical map in hand in which n double digestion fragments are placed.

Because each check needs 4 or 8 additions, let us count with the mean value 6 the operation-consume of the procedure will be $P=6C$ additions.

What is the minimum value of C? This we can get if we consider the case when $r(k)\equiv 1$. In this case C is equal (using that $n_{ab}=n_a+n_b-1$):

$$C=\sum_{k=o}^{n_{ab}-1}(n_{ab}-k)(n_{ab}+1-k)+1=$$

$$=\sum_{k=o}^{n_{ab}-1}(n_{ab}-k)^2+\sum_{k=o}^{n_{ab}-1}(n_{ab}-k)+1=$$

$$= \sum_{k=1}^{n_{ab}} k^2 + \sum_{k=1}^{n_{ab}} k+1 = \frac{n_{ab}(n_{ab}+1)(2n_{ab}+1)}{6} + \frac{n_{ab}(n_{ab}+1)}{2} + 1$$

In a special case when $n_a=6, n_b=5$, $n_{ab}=10$ C=441 and $P=2.646 \cdot 10^3$ additions.

To know the maximal value of C in this special case let us consider a very bad case using only the basic algorithm! Let us suppose that $n_a=6$, $n_b=5$, $n_{ab}=10$ and all the fragments in the double digestion are equal! In this case $r(k)=n_{ab}!/(n_{ab}-k)!$. So $C=5.602 \cdot 10^7$ and $P=3.361 \cdot 10^8$ additions. We should note that this case has only theoretical value because in practice if the fragments of the double digestion are really equal, the map of the double digestion is known. Supplying S.1. C will be 441 as at the minimal case. Or, if the fragments are only quasi-equal it is practical, to consider the fragments equal (For example if there are two fragments determined 1000 bp and 1010 bp long with 30 bp error, it is practical to consider both fragments 1005 bp long with 35 bp error); or as it is described in T.4., the algorithm will consider the continuation possibilities equal, so $r(k) \equiv 1$ as in the minimal case.

It may be expected however that, in practice, the operation-consume of the algorithm used in a common case is nearer to the lower limit than to the upper one.

If the time-consume of the program is near to the upper limit the program usually produces many possible maps. However this number and the time-consume of the program can be reduced drastically if the results of a few additional special digestions are taken into consideration.

THE USE OF THE PROGRAMS

If we have a DNA molecule, which we want to map physically, it is practical to begin the work with the enzymes which cut the DNA molecule in the fewest possible fragments. Then, having a few physical maps, we will make such double digestions in which the physical map for one of the enzymes is known. However if the fragment number is over 10 it is worthmaking a few special digestions (digestions of deletion mutants, isolated frag-

ments), because otherwise the program will give too many possibilities for the second enzyme's physical map and the procedure would be more time-consuming.

The S.2. modification increases the complexity of the algorithm so that if we can give as plus information only one fragment group containing only two fragments the use of the PMAP program is more advantagous.

If there are a few quasi-equal fragments in the double digestion, then it is more practical to determine the map in two steps. In the first step we consider the quasi-equal fragments equal (with a greater common error). So we determine the physical map variations with the program, this guides the next experimental step which is to decide among the variations, with special digestions. Knowing the physical map of two single digestions, we can determine the exact map of the double digestion considering all fragments in the double digestion as we have determined before (i.e. not equal).

EXPERIENCES IN THE USE OF THE PROGRAMS

The material used in this study was the DNA molecule of the phage 16-3 of Rhizobium meliloti 41. Details of new maps are described by Dorgai et al. (5.). Here we should mention some experiences related to the use of the programs,

The tricks described in T.1-T.7. reduced enormously the running time of the program. Here we will provide details of the trick T.5. because it may be particularly useful for the users being in contact by a parameter with the programs.

When we constructed the EcoRI map the second enzyme used was HindIII and its physical map was known. (EcoRI produces 16, HindIII 12 fragments.) From the EcoRI map the following order was known: the left cohesive end - 5 fragments in unknown order - 1 fragment - 8 fragments in unknown order - the right cohesive end. So the PMAP program solved the mapping problem in 156 CPU minutes. After this we have constructed only a beginning section not containing the HindIII-A fragment. In this situation we did not lose any information but reduced the problem with 13 unmapped fragments to a problem with 8 unmapped fragments. So the PMAP program needed half of the time.

The use of added information from special digests proved to be more useful.

When we constructed the EcoRI map we made the information from the EcoRI digestion of the HindIII-B isolated fragment available as input data. (This means the application of S.1..) In this case the PMAP program finished the procedure for the whole map in 45.44 CPU seconds instead of 156 CPU minutes.

Using the S.2. modification using PMAPS proved to be useful in practice, also. In the case of the EcoRI map PMAP gave the possible variations in 80 CPU minutes (T.5. was employed also). By putting the results of the EcoRI digest of deletion mutant A5 (this deletion touches 4 EcoRI fragments) to the input data the PMAPS program solved this problem in 16 CPU minutes. In the case of the EcoRV map (we worked with 11 EcoRV fragments with known cohesive ends and with 12 HindIII fragments with known physical map) the A5 deletion touches only 2 EcoRV fragments. The mapping problem was solved by the PMAP program in 36,5 CPU minutes and in 42,5 CPU minutes by the PMAPS program.

DISCUSSION

We have described a quick method for constructing physical maps. Using this program we have determined the physical map of the DNA molecule of the 16-3 phage of Rhizobium meliloti 41 for the enzymes EcoRI and EcoRV, which enzymes cut the DNA molecule into 16 and 11 fragments respectively (details of this and other new maps are published (5.)).

Let us compare the PMAP and PMAPS programs with the earlier ones (2.), (3.)! Unlike PMAP and PMAPS these programs cannot take into consideration the results of digests of isolated fragments and deletion mutants. However this additional informations may decrease the set of possible maps enormously. With the additional information, the PMAP and PMAPS programs are able to construct such complex physical maps which due to the time-consume cannot be done by the earlier programs. The PMAP and PMAPS programs seem to be faster in another sense, too.

Stefik illustrates the efficiency of his program (named

GA1) with the number of canonical structures of the task (2.). His published data utilize less than 1 minute of running time. The comparison of his data to the same parameters of PMAP are shown below:

GA1		PMAP	
Canonical structures	Time (seconds)	Canonical structures	Time (seconds)
5400	1.74	7200	1.56
$1.3366 \cdot 10^8$	26.8	$5.7702 \cdot 10^{16}$	45.44

We don't have enough information about the capability of the computer used by Stefik. Supposing a very pessimistic case our computer may be maximum 100 fold faster than Stefik's, but in this case a significant difference can still be detected.

Pearson's algorithm needs $P=n_a!n_b!n_{ab}\cdot(2A+M)$ operations, where A means addition and M multiplication. In our sample case, when $n_a=6$, $n_b=5$, $n_{ab}=10$ and supposing that multiplication needs double the time of addition, $P=3.456 \cdot 10^6 A$. Thus our program may be faster, but in an unfortunate case slower than Pearson's. The great advantage of our program is that it solves the fragment order of the double digestion, also. Furthermore Pearson's program is practical with the initial use of a digest of not more than 7 or 8 unmapped fragments. "If a map is well known after digestion with enzymes that cut 3,4 or 5 times, adding 8 fragment data will not be too time-consuming" (sic;2.). Our program solved this problem for 13 unmapped EcoRI fragments and for 9 unmapped EcoRV fragments. In practice the cases where Pearson's algorithm is faster are probably rare. In these cases our program gives, usually, many possibilities for the physical map. To decide which variant is the real physical map, few further targeted digestions are needed. Programming into the computer the result of these digestions will decrease drastically the time-consume, as shown in the examples presented.

The programs are written in PL/1. The programs ran on the R-40 computer of the László Kalmár Cybernetical Laboratory of the Attila József University in OS system.

Copies of the programs are available from Gábor Polner.

This work was supported by the grant 310/1981 of the

Hungarian Academy of Sciences.

ACKNOWLEDGEMENTS

We are grateful to Dr. Erzsébet Jónás, Zoltán Ascher, Anikó Páy for their help in the experimental work, to Dr. Eörs Máté for his help in the earlier stage of the theoretical work, to Dr. Attila Sára for discussing in hardware problems, to Brent Edington and Ágnes Felber for correcting the manuscript. The technical assistance of Katalin Tóth and Gréte Dömök was valuable.

REFERENCES

1. Southern, E.: Gel Electrophoresis of Restriction Fragments Methods in Enzymology, Edited by Wu, R., (1979) Vol. 68. 152-176
2. Stefik, M. (1978) Artificial Intelligence 11, 85-114
3. Pearson, W.R. (1982) Nucleic Acids Research Vol. 10. Number 1. 217-227
4. Schroeder, J.L., Blattner, F.R. (1978) Gene 4, 167-174
5. Dorgai, L., Polner, G., Jónás, E., Garamszegi, N., Ascher, Z., Páy, A., Dallmann, G., Orosz, L. (1983) Molec. gen. Genet. in press