INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 3, Number 4, Pages 481–503

PARALLEL FINITE ELEMENT METHOD FOR COUPLED CHLORIDE MOISTURE DIFFUSION IN CONCRETE

SUWITO, XIAO-CHUAN CAI, AND YUNPING XI

(Communicated by Yanping Lin)

Abstract. Penetration of chloride ions into concrete and diffusion of moisture in concrete are important factors responsible for the corrosion of steel in concrete. The two diffusion processes are coupled. This paper deals with the analysis and simulation of coupled chloride penetration and moisture diffusion in concrete. Of particular interest is the parallel programming in finite element method for solving the coupled diffusion problem. Parallel computing technology has been advantageous for solving computationally intensive problems. It has become quite mature technology and more affordable for general application. Our approach to solve the parallel programming problem is to use available libraries, i.e. Portable, Extensible Toolkit for Scientific Computation (PETSc) and The Message Passing Interface (MPI) standard. The formulation of the coupled diffusion problem, the material models involved in the differential equations, the details of parallel domain decomposition technique in the finite element algorithm are presented. The advantages of parallel programming are demonstrated by a numerical example.

Key Words. Chloride, moisture, humidity, diffusion, parallel computing, finite element, concrete.

1. Introduction

Reinforced concrete structures are often exposed to deicing salts, salt splashes, salt spray, or seawater, resulting in penetration of chloride ions into concrete. The chloride ions in the concrete will eventually reach the embedded reinforcement bars (rebar) and accumulate to a certain critical concentration level, at which the rebar begins to corrode. Since the density of corrosion product is lower than that of steel, the corrosion product occupies larger volume than the volume of steel consumed in the corrosion process. This volumetric mismatch will generate very high tensile stress in the concrete cover that may lead to concrete cracking and/or spalling.

The other necessary conditions for the rebar corrosion to take place are low pH value, and sufficient oxygen and moisture present in the rebar-concrete interface. For non-saturated concrete (e.g. the concrete not submerged in water), oxygen supply is usually not an issue. For old concrete structures, pH value of the concrete is usually much lower than new concrete (pH is about 12.5 to 13). Therefore the diffusion of moisture in the concrete is just as important as the diffusion of chloride ions in terms of the corrosion process of rebars. Although moisture is the carrier of chloride ions, the moisture and the chloride ions can diffuse in the same or opposite

Received by the editors October 27, 2004 and, in revised form, March 8, 2005.
2000 *Mathematics Subject Classification.* 35R35, 49J40, 60G40.

directions depending on the boundary (environmental) and initial conditions of a structure. When the moisture and chloride diffusions are considered as two fully coupled diffusion processes, the computational program is very complicated.

Chloride-induced rebar corrosion has been one of the major causes of deterioration in reinforced concrete structures. In some states of the U.S., chloride-induced deterioration may govern the service life of reinforced concrete structures. With the increasing acceptance of durability-based design for reinforced concrete structures, it is very important to develop both material models and computational tools that are able to accurately simulate the processes of chloride ion penetration and moisture diffusion in concrete. This study is aimed to address both aspects: development of theoretical models and computational techniques for coupled chloride and moisture diffusion in concrete.

The material models and governing partial differential equations for chloride ions penetration and moisture diffusion are time dependent and complex. Thus, various numerical approaches have been developed as effective tools to solve the problem. In this study, a finite element method is employed to solve the coupled chloride penetration and moisture diffusion equations. The space discretization is carried out based on a Galerkin procedure and time discretization based on mid-point time integration method.

The diffusion processes of moisture and chloride take place only in a thin layer of concrete structures (5 to 10 cm), and therefore, very small finite elements must be used in the diffusion analysis to deal with drastic variation of the moisture profiles and chloride concentration profiles. For large scale structures with different environmental conditions on their surfaces, the number of elements required for a durability analysis is enormous. If we want to further combine the diffusion analysis with stress analysis for the damage development due to the steel corrosion, the computational time will be escalated even at a higher rate, resulting in a computationally intensive program. With increased availability of parallel computing facilities, it becomes natural to implement parallel finite element method for the diffusion analysis. In this study, a parallel finite element program is implemented on a cluster of PCs using the Linux operating system, and the parallel architecture is classified as a distributed memory system.

The development of parallel finite element program used to be a time consuming and complicated process. Now, with the help of higher level libraries for parallel implementation such as PETSc (Portable, Extensible Toolkit for Scientific Computation), parallel finite element program can be developed in a relatively simple manner. PETSc provides a suite of data structures and routines for the scalable (parallel) solution of scientific applications modeled by partial differential equations. PETSc employs the MPI (Message Passing Interface) standard for all message-passing communication.

2. Governing differential equations and material parameters

2.1. Governing equations. We derive the governing partial differential equations for chloride penetration and moisture diffusion in concrete based on Fick's law and the mass balance equations for chloride and moisture in concrete. The two resulting equations are coupled and must be solved simultaneously. First of all, the flux of chloride ions (J_{Cl}) through a unit area of porous media depends on the gradient of chloride ions as well as the gradient of moisture, i.e.:

$$J_{Cl} = -\left(D_{Cl}\nabla C_f + \varepsilon D_H \nabla H\right), \tag{1}$$

where D_{Cl} = chloride diffusivity (cm^2/day), C_f = free chloride concentration (in gram of free chloride per gram of concrete, g/g), D_H = humidity diffusivity, H = pore relative humidity and ε = humidity gradient coefficient, which represents the coupling effect of moisture diffusion on chloride penetration. In some literatures on transport properties of concrete, D_{Cl} is called absolute diffusivity, and εD_H is called relative diffusivity.

The moisture content in concrete can be described by water content (w) or by the relative humidity in pores (H), called pore relative humidity. The problem with water content is that even if the total water content is fixed (for a completely sealed concrete sample), the mobile water content is a function of time because a part of the water is consumed by the hydration reaction and thus becomes chemically combined water, which is immobile. Therefore, it is advantageous to use pore relative humidity to represent moisture content, which is a combined indicator for liquid water and water vapor [9]. In the present study, the moisture flux (J_H) is related to the gradients of chloride and pore relative humidity, i.e.:

$$J_H = -\left(\delta D_{Cl}\nabla C_f + D_H \nabla H\right) \tag{2}$$

in which δ = chloride gradient coefficient, which represents the coupling effect of chloride ions on moisture diffusion.

The mass balance of chloride ions and moisture can be expressed using Fick's second law as follows:

$$\frac{\partial C_t}{\partial C_f}\frac{\partial C_f}{\partial t} = -\nabla \cdot [J_{Cl}] \tag{3}$$

$$\frac{\partial w}{\partial H}\frac{\partial H}{\partial t} = -\nabla \cdot [J_H]\,, \tag{4}$$

where $\partial C_t/\partial C_f$ is the chloride binding capacity and $\partial w/\partial H$ is the moisture capacity; C_t is the total chloride concentration (in gram of free chloride per gram of concrete, g/g). When chloride ions enter the concrete, some of them attach to the pore wall, become bound chloride; and some of them are free to diffuse around. So, only the free chloride is considered in Eq. (1). The binding capacity of chloride in Eq. (3) is the ratio of the total chloride and the free chloride. Similarly, the moisture capacity represents the bound moisture on the surface of pore wall. Substituting Eq. (1) into Eq. (3) and Eq. (2) into Eq. (4) give:

$$\frac{\partial C_t}{\partial C_f}\frac{\partial C_f}{\partial t} = \nabla \cdot [(D_{Cl}\nabla C_f + \varepsilon D_H \nabla H)] \tag{5}$$

$$\frac{\partial w}{\partial H}\frac{\partial H}{\partial t} = \nabla \cdot [(\delta D_{Cl}\nabla C_f + D_H \nabla H)]\,, \tag{6}$$

which can be rewritten as:

$$\frac{\partial C_t}{\partial C_f}\frac{\partial C_f}{\partial t} = \nabla \cdot [D_{Cl}\nabla C_f + D_\varepsilon \nabla H] \tag{7}$$

$$\frac{\partial w}{\partial H}\frac{\partial H}{\partial t} = \nabla \cdot [D_\delta \nabla C_f + D_H \nabla H] \tag{8}$$

in which $D_\varepsilon = \varepsilon D_H$ and $D_\delta = \delta D_{Cl}$. Eqs. (7) and (8) are the governing equations for the coupled problem of chloride ions and moisture diffusions. The general boundary conditions are

$$C_f = C_0 \qquad \text{on} \qquad \Gamma_1 \tag{9}$$

$$D_{Cl}\frac{\partial C_f}{\partial n} + J_{Cl} + D_\varepsilon \frac{\partial H}{\partial n} + \alpha_{Cl}\left(C_f - C_{fa}\right) = 0 \qquad \text{on} \qquad \Gamma_2 \tag{10}$$

$$C_f = C_0 \qquad \text{on} \qquad \Gamma_3 \tag{11}$$

$$D_H \frac{\partial H}{\partial n} + J_H + D_\delta \frac{\partial C_f}{\partial n} + \alpha_H\left(H - H_a\right) = 0 \qquad \text{on} \qquad \Gamma_4, \tag{12}$$

where α_{Cl} and α_H are the convective chloride and relative humidity coefficients; and C_{fa} and H_a are ambient chloride ions and relative humidity; respectively. Γ_1 and Γ_3 are the part of boundary with a constant chloride ions and relative humidity; and Γ_2 and Γ_4 are the part of boundary subjected to a specified chloride ions and relative humidity flux; respectively. Γ_1 and Γ_2 form the complete boundary surface for the chloride ions diffusion problem, and Γ_3 and Γ_4 for the moisture diffusion problem.

It is important to note that the present formulation is different from the previous formulation (e.g. [2]) in that Eqs. (7) and (8) are two fully coupled (two-way coupled) partial differential equations, while the previous formulation was one-way coupled equations.

2.2. Material parameters. In Eqs. (1) through (12), there are many material parameters. It is very important to use reliable material models in the equations in order to obtain an accurate prediction of the coupled diffusion processes. The six material parameters in Eqs. (7) and (8) can be divided into three groups. $\partial C_t/\partial Cf$ and $\partial w/\partial H$ are binding capacities, D_ε and D_δ are coupling parameters, and D_{Cl} and D_H are diffusivities. In the following, the material models used in the present study will be briefly described.

A recently developed model for the chloride binding capacity [33] will be used, i.e.:

$$\frac{dC_f}{dC_t} = \frac{1}{1 + \frac{A\,10^B \beta_{C-S-H}}{35450\beta_{sol}}\left(\frac{C_f}{35.45\beta_{sol}}\right)^{A-1}}, \tag{13}$$

where A and B are two material constants related to chloride adsorption and equal to 0.3788 and 1.14, respectively [27]. The binding capacity depends on the two parameters, β_{sol} and β_{C-S-H}. The parameter β_{sol} is the ratio of pore solution to concrete (L/g) and the parameter β_{C-S-H} is the ratio of C-S-H gel (calcium silicate hydrate gel) to concrete (g/g). The detail of derivations of β_{sol} and β_{C-S-H} can be found in the paper by Xi and Bazant [32].

The diffusivity of chloride ions in concrete can be estimated using the multifactor method as follows:

$$D_{Cl} = f_1 f_2(g_i) f_3(H) f_4(T) f_5(C_f), \tag{14}$$

where f_1 is a factor accounting for the influential parameters and resulted from the calibration of the numerical model with the experimental data. Since the diffusivity of concrete is mainly influenced by its water-cement ratio (w/c) and curing age (t_0), an expression for the factor f_1 was suggested:

$$f_1 = \frac{28 - t_0}{62500} + \left(\frac{1}{4} + \frac{(28 - t_0)}{300}\right)(w/c)^{6.55}. \tag{15}$$

The second factor $f_2(g_i)$ is to account for the effect of composite action of the aggregates and the cement paste on the diffusivity of concrete. This factor can be

formulated using the three phase composite model developed by Christensen [13]:

$$f_2(g_i) = D_{cp}\left(1 + \frac{g_i}{[1-g_i]/3 + 1/[(D_{agg}/D_{cp}) - 1]}\right) \tag{16}$$

in which g_i is the volume fraction of aggregates in the concrete, D_{agg} and D_{cp} are the diffusivities of aggregates and cement paste. The diffusivity of the aggregates and cement paste can be calculated using the model proposed by Martys et al. [19]:

$$D = \frac{2\left(1 - \left(V_p - V_p^c\right)\right)}{S^2}\left(V_p - V_p^c\right)^{4.2}, \tag{17}$$

where V_p is the porosity, S is the surface area and V_p^c is the critical porosity (the porosity at which the pore space is first percolated). Numerical simulation of porous materials composed of randomly overlapping spheres showed that the pore space becomes disconnected at $V_p^c = 3\%$ (Martys et al. [19]). The value of D_{agg} can be evaluated by Eq. (17) (if S, V_p, and V_p^c for the aggregate are known) or may be taken as a constant (the approach taken in this study), typically 1 x 10^{-12}cm/s. D_{cp} can be evaluated using Eq. (13), in which the surface area of cement paste, S, can be estimated by the monolayer capacity, V_m, of adsorption isotherm of concrete since V_m is proportional to S (Xi et al. [33, 34]; Xi [30, 31]). The porosity, V_p, can be estimated by absorption isotherm, $n(H,T) = W_{sol}/W_{conc}$ at saturation ($H = 1$). W_{sol} and W_{conc} are the weight of pore solution and concrete, respectively. The detail treatment of absorption isotherm can be seen on papers by Xi et al. [33, 34] and Xi [30, 31].

The third factor, $f_3(H)$ is to account for the effect of relative humidity level on the chloride diffusivity. A model proposed by Bazant and Najjar [9] can be used, which was developed initially for moisture diffusion. In this paper, assuming the analogy between the moisture and the chloride diffusion, the model is:

$$f_3(H) = \left(1 + \frac{(1-H)^4}{(1-H_C)^4}\right)^{-1} \tag{18}$$

in which H_c is the critical humidity level at which the diffusivity drops halfway between its maximum and minimum values ($H_c = 0.75$).

The fourth factor, $f_4(T)$, is to account for the effect of temperature on the diffusivity of concrete. This can be done by using Arrhenius' law as follows

$$f_4(T) = \exp\left[\frac{U}{R}\left(\frac{1}{T_0} - \frac{1}{T}\right)\right] \tag{19}$$

in which U is the activation energy of the diffusion process, R is the gas constant (8.314 J mol^{-1}K^{-1}), T and T_0 are the current and reference temperatures, respectively, in Kelvin ($T_0 = 296$ K). U has been found to depend on water-to-cement ration, w/c, and the cement type [14, 21] (see Table 1).

TABLE 1. Activation energies for various cement paste

w/c	**Ordinary portland cement (KJ/mol)**	**Cement with pozzolans (KJ/mol)**
0.4	41.8 ± 4.0	–
0.5	41.8 ± 4.0	4.18
0.6	41.8 ± 4.0	–

The fifth factor, $f_5(C_f)$, is to account for the dependence of the chloride diffusivity on the free chloride concentration, called concentration dependence

$$f_5(C_f) = 1 - k_{ion}(C_f)^m \tag{20}$$

in which k_{ion} and m are constants, 8.333 and 0.5, respectively [32].

Considering concrete as two-phase material consisting of cement paste and aggregate, the moisture capacity of concrete can be evaluated by taking the average of the moisture capacities of cement paste and aggregate as follows:

$$\frac{dw}{dH} = f_{agg}\left(\frac{dw}{dH}\right)_{agg} + f_{cp}\left(\frac{dw}{dH}\right)_{cp} \tag{21}$$

in which f_{agg} and f_{cp} are the weight percentages of the aggregate and cement paste, respectively; $(dw/dH)_{agg}$ and $(dw/dH)_{cp}$ are the moisture capacities of aggregate and cement paste, respectively; and water content, w, is given by the BET equation:

$$w = \frac{CkV_mH}{(1-kH)\left[1+(C-1)\,kH\right]}, \tag{22}$$

where V_m, C and k are constants evaluated by curve fitting of the adsorption test data. The details for evaluating the Eqs. (21) and (22) can be seen in [30, 31, 33, 34].

The moisture diffusivity of concrete can be predicted by the composite action as follows [13]:

$$D_H = D_{Hcp}\left(1 + \frac{g_i}{[1-g_i]/3 + 1/[(D_{Hagg}/D_{Hcp}) - 1]}\right) \tag{23}$$

in which g_i is the aggregate volume fraction, D_{Hcp} is the humidity diffusivity of the cement paste and D_{Hagg} is the humidity diffusivity of the aggregates.

The humidity diffusivity of aggregates in concrete is very small due to the fact that the pores in aggregates are discontinuous and enveloped by cement paste and so it can be neglected. The humidity diffusivity of cement paste can be predicted by the following empirical formula:

$$D_{Hcp} = \alpha_h + \beta_h[1 - 2^{-10^{\gamma_h(H-1)}}], \tag{24}$$

where α_h, β_h and γ_h are coefficients to be calibrated from the test data and given as:

$$\alpha_h = 1.05 - 3.8(w/c) + 3.56(w/c)^2 \tag{25}$$

$$\beta_h = -14.4 + 50.4(w/c) - 41.8(w/c)^2 \tag{26}$$

$$\gamma_h = 31.3 - 136(w/c) + 162(w/c)^2. \tag{27}$$

So far, there have been no material models developed for the two coupling parameters, D_ε and D_δ. Most recently, Ababneh and Xi conducted some experimental studies for the coupling effect of chloride concentration on moisture diffusion [3]. Ababneh [1] also conducted an experimental study on the effect of moisture concentration on chloride diffusion. In the present study, we used the available test data to determine the two constants ε and δ, and then determine the two coupling parameters. The specific values of ε and δ are given in Table 2.

TABLE 2. Material parameters used in the analysis

Parameter	Value
Water to Cement Ration (w/c)	0.55
Volume Fraction of Aggregate	0.65
Cement Type	I
ε	0.028
δ	0.2

3. Finite element formulation

Before the parallel implementation of the coupled governing equations, we need to briefly introduce the finite element formulation, which provides the basis for the discussion in the next sections. The continuous variables in the coupled chloride penetration and moisture diffusion equations, free chloride C_f and moisture H_m, are spatially discretized over the space domain, Ω. The discretization process can be expressed as follows

$$\Omega = \bigcup_{e=1}^{nelem} \Omega^e, \tag{28}$$

where *nelem* is the total number of finite elements and Ω^e is a finite element. Also we define $\partial\Omega$ as the boundary of the problem domain and $\partial\Omega^e$ the boundary of a finite element.

Using isoparametric elements, the unknowns in the coupled chloride penetration and moisture diffusion equations, C_f and H_m, are approximated in terms of nodal values $\{\hat{\boldsymbol{C}}_f\}$ and $\{\hat{\boldsymbol{H}}_m\}$, respectively, as

$$C_f \approx \lfloor \boldsymbol{N}_c \rfloor \left\{ \hat{\boldsymbol{C}}_f \right\} \tag{29}$$

$$H_m \approx \lfloor \boldsymbol{N}_h \rfloor \left\{ \hat{\boldsymbol{H}}_m \right\}, \tag{30}$$

where $\lfloor \boldsymbol{N}_c \rfloor$ and $\lfloor \boldsymbol{N}_h \rfloor$ are the element shape functions for C_f and H_m, respectively. The notations $\lfloor\rfloor$ and $\{\}$ are for row and column vectors, respectively. The element shape functions are defined as

$$\lfloor \boldsymbol{N}_c \rfloor = \lfloor \boldsymbol{N}_h \rfloor \equiv \lfloor N_1 \;\; N_2 \;\; \cdots \;\; N_n \rfloor \tag{31}$$

in which N_i is the shape function for node i and n is the total number of nodes of an element. The vectors $\left\{ \hat{\boldsymbol{C}}_f \right\}$ and $\left\{ \hat{\boldsymbol{H}}_m \right\}$ can be expressed as

$$\left\{ \hat{\boldsymbol{C}}_f \right\} \equiv \left\lfloor \hat{C}_{f1} \;\; \hat{C}_{f2} \;\; \cdots \;\; \hat{C}_{fn} \right\rfloor^T \tag{32}$$

$$\left\{ \hat{\boldsymbol{H}}_m \right\} \equiv \left\lfloor \hat{H}_{m1} \;\; \hat{H}_{m2} \;\; \cdots \;\; \hat{H}_{mn} \right\rfloor^T, \tag{33}$$

where $\hat{C}_{fi}$ and $\hat{H}_{mi}$ are the value of C_f and H_m, respectively, at node i.

Substituting the approximated values of Eqs. (29) and (30) into Eqs. (7) and (8), respectively, and applying the Galerkin procedure result in

$$\frac{d}{dt}\left(\left[\boldsymbol{C}_e(\hat{\boldsymbol{\phi}}) \right] \left\{ \hat{\boldsymbol{\phi}} \right\} \right) = \left[K_e(\hat{\boldsymbol{\phi}}) \right] \left\{ \hat{\boldsymbol{\phi}} \right\}, \tag{34}$$

where the element matrices $\boldsymbol{K}_e$ and $\boldsymbol{C}_e$ are expressed as follows

$$[\boldsymbol{K}_e] = \begin{bmatrix} \boldsymbol{K}_{cc} & \boldsymbol{K}_{ch} \\ \boldsymbol{K}_{hc} & \boldsymbol{K}_{hh} \end{bmatrix} \tag{35}$$

$$[\boldsymbol{K}_{cc}] = -\int_{\Omega_e} \nabla \lfloor \boldsymbol{N}_c \rfloor^T D_{Cf} \nabla \lfloor \boldsymbol{N}_c \rfloor \, d\Omega + \int_{\partial\Omega_e} \lfloor \boldsymbol{N}_c \rfloor^T D_{Cf} \nabla \lfloor \boldsymbol{N}_c \rfloor \, d\Gamma \tag{36}$$

$$[\boldsymbol{K}_{ch}] = -\int_{\Omega_e} \nabla \lfloor \boldsymbol{N}_c \rfloor^T D_{\varepsilon} \nabla \lfloor \boldsymbol{N}_h \rfloor \, d\Omega + \int_{\partial\Omega_e} \lfloor \boldsymbol{N}_c \rfloor^T D_{\varepsilon} \nabla \lfloor \boldsymbol{N}_h \rfloor \, d\Gamma \tag{37}$$

$$[\boldsymbol{K}_{hc}] = -\int_{\Omega_e} \nabla \lfloor \boldsymbol{N}_h \rfloor^T D_{\delta} \nabla \lfloor \boldsymbol{N}_c \rfloor \, d\Omega + \int_{\partial\Omega_e} \lfloor \boldsymbol{N}_h \rfloor^T D_{\delta} \nabla \lfloor \boldsymbol{N}_c \rfloor \, d\Gamma \tag{38}$$

$$[\boldsymbol{K}_{hh}] = -\int_{\Omega_e} \nabla \lfloor \boldsymbol{N}_h \rfloor^T D_{Hm} \nabla \lfloor \boldsymbol{N}_h \rfloor \, d\Omega + \int_{\partial\Omega_e} \lfloor \boldsymbol{N}_h \rfloor^T D_{Hm} \nabla \lfloor \boldsymbol{N}_h \rfloor \, d\Gamma \tag{39}$$

$$[\boldsymbol{C}_e] = \begin{bmatrix} \boldsymbol{C}_c & 0 \\ 0 & \boldsymbol{C}_h \end{bmatrix} \tag{40}$$

$$[\boldsymbol{C}_c] = \int_{\Omega_e} \lfloor \boldsymbol{N}_c \rfloor^T c_c \lfloor \boldsymbol{N}_c \rfloor \, d\Omega \tag{41}$$

$$[\boldsymbol{C}_h] = \int_{\Omega_e} \lfloor \boldsymbol{N}_h \rfloor^T c_h \lfloor \boldsymbol{N}_h \rfloor \, d\Omega \tag{42}$$

and the vector $\left\{\hat{\boldsymbol{\phi}}\right\}$ is defined as

$$\left\{\hat{\boldsymbol{\phi}}\right\} = \left\lfloor \hat{\boldsymbol{C}}_f \;\; \hat{\boldsymbol{H}}_m \right\rfloor^T . \tag{43}$$

Finally, applying the mid-point integration method to Eq. (34) over the time interval $[t^{\xi+1}, t^{\xi}]$ results in

$$\left(\left[\boldsymbol{C}_e(\hat{\boldsymbol{\phi}}) - \theta \Delta t \, \boldsymbol{K}_e(\hat{\boldsymbol{\phi}})\right] \left\{\hat{\boldsymbol{\phi}}\right\}\right)^{\xi+1} = \left(\left[\boldsymbol{C}_e(\hat{\boldsymbol{\phi}}) + (1-\theta) \, \Delta t \, \boldsymbol{K}_e(\hat{\boldsymbol{\phi}})\right] \left\{\hat{\boldsymbol{\phi}}\right\}\right)^{\xi}, \tag{44}$$

where $\Delta t \equiv t^{\xi+1} - t^{\xi}$ in which t^{ξ} is the time at time step ξ. The value of parameter θ is related to the solution method adopted in the program. Typical values of θ are 0, $1/2$ and 1 that correspond to fully explicit, semi-implicit and fully implicit methods, respectively. The semi-implicit method is used in this study. Eq. (44) is ready for software implementation, in which all the known values are on the right hand side and all the unknown values are on the left hand side of equation. Eq. (44) can be further simplified to

$$[\boldsymbol{A}]^{\xi+1} \left\{\hat{\boldsymbol{\phi}}\right\}^{\xi+1} = \{\boldsymbol{b}\}^{\xi} \tag{45}$$

$$[\boldsymbol{A}]^{\xi+1} = \left[\boldsymbol{C}_e(\hat{\boldsymbol{\phi}}) - \theta \Delta t \, \boldsymbol{K}_e(\hat{\boldsymbol{\phi}})\right]^{\xi+1} \tag{46}$$

$$\{\boldsymbol{b}\}^{\xi} = \left(\left[\boldsymbol{C}_e(\hat{\boldsymbol{\phi}}) + (1-\theta) \, \Delta t \, \boldsymbol{K}_e(\hat{\boldsymbol{\phi}})\right] \left\{\hat{\boldsymbol{\phi}}\right\}\right)^{\xi} . \tag{47}$$

4. Parallel implementation of finite element program

The purpose of developing a parallel computing program is to make a significant reduction in the executing time of the program. Ideally, if the executing time for solving a problem is T on a one-processor system, then, with a p-processor system, the executing time would be simply T/p. But, one must realize that the ideal condition only applies to a very specific problem: the problem that can be evenly distributed within a number of processors without the need of any communication between the processors. Apparently, the ideal problem almost does not exist. In reality, a properly designed parallel program can achieve a significant reduction in the executing time. The efficiency and effectiveness of the parallel program depend largely on several parameters, such as the problem to be solved with selected algorithms, the parallel paradigms and hardware architectures.

The parallel implementation of the finite element program developed in this study is based on the distributed memory. In the distributed memory system, each processor has its own memory module. Such a distributed memory system is built by connecting each component with a high-speed communications network. Individual processor communicates to each other over the network. Inter-processor communications are managed by a message-passing technology. The industry standard interface for message passing is The Message Passing Interface (MPI) standard, which is a library callable from C or Fortran.

Developing parallel program using MPI directly is quiet complex. Thus, PETSc is used in this study, which is a higher level library than MPI and it provides a suite of data structures and routines for the scalable (parallel) solution of scientific applications modeled by partial differential equations. PETSc employs the MPI standard for all message-passing communication and hides lower level programming implementation.

4.1. Parallel finite element algorithm. In general, there are two strategies that can be used for parallel implementation of a finite element program, i.e.: (1) the system equations are formed by the usual sequential approach (without any partitioning of problem domain), but solved in parallel; and (2) the problem domain is divided into a number of sub-domains and then the system equations are formed concurrently for each sub-domain and solved also in parallel. The first approach, also called implicit domain decomposition approach, is generally suitable for static or steady state and linear problems; while the second approach, called the explicit domain decomposition approach, is more suitable for time dependent and nonlinear problems. This is because the computation time of time dependent and nonlinear problems is more dominant than that of static and linear problems. Since the problem at hand is a time dependent problem, the second approach is employed.

A parallel program is typically developed by dividing the program into multiple fragments that can execute simultaneously, each on its own processor. In the finite element analysis, this can be accomplished by applying a domain decomposition method [25]. Domain decomposition method is the method usually used for solving large scale system equations and it is also suitable for parallel programming because of data locality. There are two types of domain decomposition methods, overlapping and non-overlapping methods. The non-overlapping method, which is also known as sub-structuring method, is employed in this study.

In the non-overlapping domain decomposition method, after generating mesh, $\mathcal{T}$, in the domain, Ω, the mesh is then partitioned into non-overlapping sub-mesh,

$\mathcal{T}_i$. The process can be mathematically expressed as

$$\mathcal{T} = \cup_i \mathcal{T}_i \tag{48}$$

and

$$\mathcal{T}_i \cap \mathcal{T}_j = \{\} \quad \text{for} \quad i \neq j. \tag{49}$$

Once the mesh is partitioned, the contributions of the system of equation (Eq. (45)) from individual sub-mesh can be assembled concurrently. The assembled system equation can be written in the block matrix form as

$$\begin{bmatrix} \boldsymbol{A}_{II}^1 & & & \boldsymbol{A}_{IB}^1 \\ & \ddots & & \vdots \\ & & \boldsymbol{A}_{II}^p & \boldsymbol{A}_{IB}^p \\ \boldsymbol{A}_{BI}^1 & \cdots & \boldsymbol{A}_{BI}^p & \boldsymbol{A}_{BB} \end{bmatrix} \begin{Bmatrix} \hat{\boldsymbol{\varphi}}_I^1 \\ \vdots \\ \hat{\boldsymbol{\varphi}}_I^p \\ \hat{\boldsymbol{\varphi}}_B \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{b}_I^1 \\ \vdots \\ \boldsymbol{b}_I^2 \\ \boldsymbol{b}_B \end{Bmatrix}, \tag{50}$$

where subscript "I" and "B" correspond to the internal and boundary degree of freedoms, respectively. Blocks $\boldsymbol{A}_{BB}$ and $\boldsymbol{b}_B$ have the contribution from all sub-domains, can be expressed as

$$\boldsymbol{A}_{BB} = \sum_{i=1}^{p} \boldsymbol{A}_{BB}^i \tag{51}$$

$$\boldsymbol{b}_B = \sum_{i=1}^{p} \boldsymbol{b}_B^i \tag{52}$$

for $i = 1, \ldots, p$. It should be noticed that the following block components

$$\begin{bmatrix} \boldsymbol{A}_{II}^i & \boldsymbol{A}_{IB}^i \\ \boldsymbol{A}_{BI}^i & \boldsymbol{A}_{BB}^i \end{bmatrix} \quad \text{and} \quad \begin{Bmatrix} \boldsymbol{b}_I^i \\ \boldsymbol{b}_B^i \end{Bmatrix} \tag{53}$$

can be computed entirely on processor i, which works only on sub-domain Ω_i. This means that all processors can work concurrently to form their own block components as expressed in Eq. (53). Eliminating all internal unknowns ($\hat{\boldsymbol{\varphi}}_I^i$) from Eq. (50) results in a Schur complement system as expressed

$$\boldsymbol{A}_{sc} \hat{\boldsymbol{\varphi}}_B = \boldsymbol{b}_{sc} \tag{54}$$

with

$$\boldsymbol{A}_{sc} = \sum_{i=1}^{p} \left(\boldsymbol{A}_{BB}^i - \boldsymbol{A}_{BI}^i \left(\boldsymbol{A}_{II}^i \right)^{-1} \boldsymbol{A}_{IB}^i \right) \tag{55}$$

and

$$\boldsymbol{b}_{sc} = \boldsymbol{b}_B - \sum_{i=1}^{p} \left(\boldsymbol{A}_{BI}^i \left(\boldsymbol{A}_{II}^i \right)^{-1} \boldsymbol{b}_I^i \right). \tag{56}$$

Solving Eq. (54) directly by first forming the Schur complement matrix ($\boldsymbol{A}_{sc}$) could be computationally expensive because of the need to form matrix $\boldsymbol{A}_{sc}$ from each substructure. The size of matrix $\boldsymbol{A}_{sc}$ grows quite fast as the number of substructures grows. Therefore, instead of solving Eq. (54) directly, an iterative domain decomposition method will be employed, in which the reduced system (Eq. (54)) is solved without explicitly forming the Schur complement matrix. Each sub-domain is responsible for providing the matrix-vector product, $\boldsymbol{A}_{sc}^i \hat{\boldsymbol{\varphi}}_B^i$, and the modified right hand side vector, $\boldsymbol{b}_{sc}^i$. The procedure for computing $\boldsymbol{A}_{sc}^i \hat{\boldsymbol{\varphi}}_B^i$ involves the following steps:

- Compute $\boldsymbol{y}_1 = \boldsymbol{A}_{BB}^i \hat{\boldsymbol{\varphi}}_B^i$

- Compute $\boldsymbol{w} = \boldsymbol{A}^i_{IB}\hat{\boldsymbol{\varphi}}^i_B$
- Solve $\boldsymbol{A}^i_{II}\boldsymbol{z} = \boldsymbol{w}$ for $\boldsymbol{z}$
- Compute $\boldsymbol{y}_2 = \boldsymbol{A}^i_{BI}\boldsymbol{z}$
- Compute $\boldsymbol{y}_1 - \boldsymbol{y}_2$.

As for the modified right hand side vector, b^i_{sc}, the computation procedures are:

- Solve $\boldsymbol{A}^i_{II}\boldsymbol{s} = \boldsymbol{b}^i_I$ for $\boldsymbol{s}$
- Compute $\boldsymbol{q} = \boldsymbol{A}^i_{BI}\boldsymbol{s}$
- Compute $\boldsymbol{b}^i_{sc} = \boldsymbol{b}^i_B - \boldsymbol{q}$.

Once the iterative procedure converges, the internal unknowns for each sub-domain, $\hat{\boldsymbol{\varphi}}^i_I$, can be obtained by solving the following equation

$$\boldsymbol{A}^i_{II}\hat{\boldsymbol{\varphi}}^i_I = \boldsymbol{b}^i_I - \boldsymbol{A}^i_{IB}\hat{\boldsymbol{\varphi}}^i_B. \tag{57}$$

Iterative solver must be used in the iterative domain decomposition method. For this study, the iterative solver GMRES (Generalized Minimal Residual method) (see Saad [22]) was chosen. GMRES is mainly chosen because of its ability to solve nonsymmetric linear system as in the case of our problem.

To improve the convergence of the interface problem (Eq. (54)), the preconditioning will be applied. Since the Schur complement matrix $\boldsymbol{A}_{sc}$ is never assembled, the Jacobi preconditioning (i.e. using the diagonal part of $\boldsymbol{A}_{sc}$) provided by PETSc cannot be used. Thus for this study, a simple preconditioning will be formed by taking the diagonal parts of the interface matrix $\boldsymbol{A}_{BB}$. This preconditioning performed relatively well for our study, as shown by the number of iterations needed to solve the Schur complement system iteratively (Table 3). It is obvious that the number of iterations increase with the increasing problem sizes, but the increasing number of iteration tends to reach the certain maximum number.

TABLE 3. Number of iterations for solving Schur complement system

# of Procs	# of Nodes	Time Steps			
		1	1000	3000	5000
2	3000	10	11	11	11
	12000	16	18	19	19
4	3000	10	10	11	12
	12000	15	17	18	18
8	3000	11	12	13	13
	12000	16	17	18	18
16	3000	11	12	12	13
	12000	16	17	18	19
32	3000	11	12	12	13
	12000	17	18	19	20
64	3000	11	12	13	13
	12000	16	19	20	20

4.2. Parallel programming. The parallel finite element program is developed based on a single program multiple data (SPMD) model of parallel execution, in which each processor follows the same algorithm but operates on a different

sub-mesh. Following the SPMD model, the finite element mesh is partitioned, preprocessed and stored in a number of input files, one file for each sub-mesh. The number of input files is the same as the number of processors.

The linear triangular element is used in this study. The triangular mesh is generated using the free software called Triangle [23, 24]. For partitioning the mesh, the METIS software library [17] developed by the University of Minnesota is employed. METIS provides the tools to efficiently partition unstructured mesh while evenly distributing the computation load and minimizing the total communication among processors. Once the mesh has been partitioned, they need to be processed by a preprocessor program so that each sub-mesh consisting of a certain number of nodes and elements followed by a list of nodes and a list of element connectivities and also followed by the information of nodes that are shared with neighboring sub-meshes and the global degree of freedoms of those nodes. Fig. 1 shows an example of the mesh numbering scheme.

The previously described algorithm was implemented using PETSc, which provides algebraic data objects such as vector and matrix, routines for managing vector and matrix and numerical libraries such as solvers and preconditioners. In addition, PETSc also includes the routines for managing parallel data layout such as mapping between the global numberings, local and global indices, scattering, etc. Thus, PETSc can be considered as a toolkit that can ease the development of parallel finite element program and consequently reduce its development time.

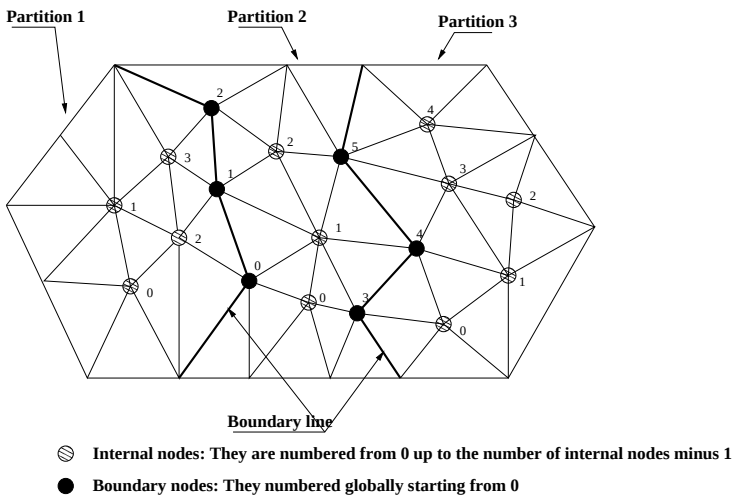

FIGURE 1. An example renumbering of internal and boundary nodes

PETSc library can be called using C, C++ and FORTRAN languages, with some limitations using FORTRAN because of the language syntax. Regardless of programming language used, a program developed using PETSc has three elements in common: (1) including one or more PETSc header files; (2) initializing PETSc; and (3) finalizing PETSc.

After the initialization of PETSc, the program spawns into the number of processes as specified by the user. Each input file can then be opened and read by its processor. After reading all data from the input file, each processor starts the computing process for each time step. First, the data structures for vectors and matrices needed for computation must be prepared. Based on Eq. (53), four

sequential (not parallel) matrices ($\boldsymbol{A}^i_{II}$, $\boldsymbol{A}^i_{IB}$, $\boldsymbol{A}^i_{BI}$ and $\boldsymbol{A}^i_{BB}$) and two sequential vectors ($\boldsymbol{b}^i_I$ and $\boldsymbol{b}^i_B$) are needed.

Using the PETSc objects for matrices and vectors, we can form all components in Eq. (53) for each processor. A direct approach would be then to form the matrix $\boldsymbol{A}_{sc}$ and $\boldsymbol{b}_{sc}$ and solve Eq. (54) for $\hat{\boldsymbol{\varphi}}_B$. However, the direct approach is considered inefficient because the matrix $\boldsymbol{A}_{sc}$ is expensive to be assembled and usually dense. Thus, an iterative domain decomposition method without actually forming matrix $\boldsymbol{A}_{sc}$ was employed. The iterative domain decomposition method can be accomplished in PETSc by using the matrix-free method combined with the iterative solver. The matrix-free method creates the matrix structure without actually generating the matrix and must be solved using the iterative solver. Also it is possible to use our own preconditioner in the matrix-free method.

PETSc provides a wide range of solvers, both direct and iterative methods. In this study, the LU factorization (direct method) is used to solve the local problem and the GMRES global problem. PETSc also provides a number of preconditioners, but, because of the matrix-free method, we cannot use those preconditioners. Instead, we need to implement our own preconditioner. Fortunately, PETSc has a routine that facilitates the implementation of our own preconditioner.

Also, since PETSc uses MPI standard for communication between processors, the program is portable for most computer systems.

4.3. Performance of parallel finite element implementation. Some comparative parameters are needed to measure the performance of the parallel implementation of finite element program. Two commonly used parameters are speed-up and efficiency. The speed-up can be defined as the ratio between the time taken by the code to execute on a single processor, $T_s(n)$, and the time taken for the same code to execute on p processors, $T_p(n)$,

$$S_p(n) = \frac{T_s(n)}{T_p(n)}, \tag{58}$$

where n is the problem size. Parallel efficiency can be expressed as the ratio between speed-up and the number of processors,

$$E_p(n) = \frac{S_p(n)}{p}. \tag{59}$$

There are some different scenarios on how to interpret $T_s(n)$ in Eq. (58). Ideally, the $T_s(n)$ would be the time taken to run the fastest serial algorithm on the fastest serial machine available. However, in practice, most researchers do not have the fastest machine. Instead, they usually take $T_s(n)$ as the time taken to run the fastest serial algorithm on the fastest processor in their parallel computing systems.

Another possibility is to take $T_s(n)$ as the time taken to run the parallel algorithm on one processor. This is usually done for examining the performance of the parallel algorithm. However, not all parallel algorithms is able to run on one processor as in our parallel algorithm which will only run on two or more processors. To evaluate the performance of a parallel algorithm for the case in which the parallel algorithm cannot be run in one processor, one may replace $T_s(n)$ with $T_2(n)$, the time taken to run the parallel algorithm in two processors.

5. Comparison with test data

Experimental results of 90-day ponding test conducted by Andrade and Whiting [6] are compared with numerical results from our model. As shown in Fig. 2,

for $w/c = 0.6$, the numerical result agrees well with experimental data. When the water-cement ratio is low, i.e. $w/c = 0.4$, the model prediction is not accurate. This is due to the fact that admixtures (such as water reducing admixture) are used in concrete with low w/c to make the concrete mix workable, while the material models used in this study did not include the effects of admixtures.

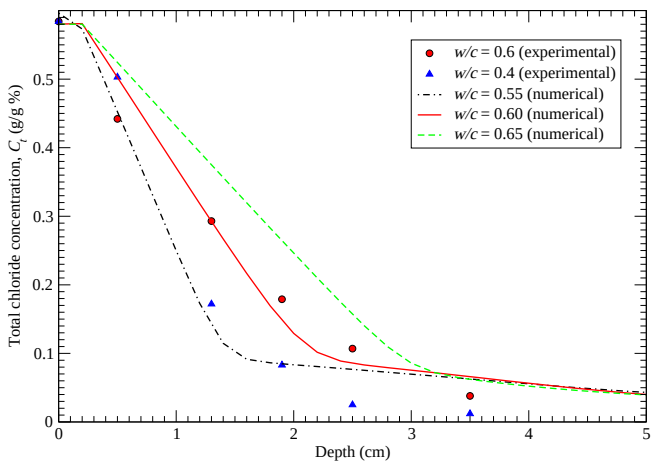

FIGURE 2. Comparison with experimental results

6. A numerical example

The physical model used in the present study is shown in Fig. 3. It is a concrete slab of 15 cm by 30 cm. The concrete slab contains initially no chloride ions and it has 60% relative humidity (RH). The concrete slab is exposed to 3% NaCl and 100% RH on the top surface. The other boundaries are assumed to be sealed. Obviously, this example is not to simulate the concrete under service condition, but to simulate the commonly used long-term pond test. With these initial and boundary conditions, one can speculate that the chloride and the moisture diffusions are in the same direction. The material parameters used in the example are shown in Table 2.

Fig. 4 shows the typical finite element mesh used in the analysis and Fig. 5 shows the typical mesh partitioned into 8 sub-meshes. The mesh is partitioned so that each sub-mesh has almost the same number of elements, and that the number of nodes on sub-mesh boundaries is as small as possible.

Figs. 6-9 show numerical results from the coupled chloride and moisture diffusion analysis. Fig. 6 shows the distribution of chloride ions in the concrete after 400 days and Fig. 7 the distribution of relative humidity after 400 days. One can see from the color contours in Fig. 6 and Fig. 7 that the chloride ions and moisture are penetrating in the same direction towards the bottom of the concrete slab.

Fig. 8. shows the chloride profile as a function of time at different depths, i.e. at 1 cm, 3 cm and 5 cm. If we chose the critical chloride ion content of 0.04% and assume that there is a rebar at the depth of 5 cm from the top surface of the

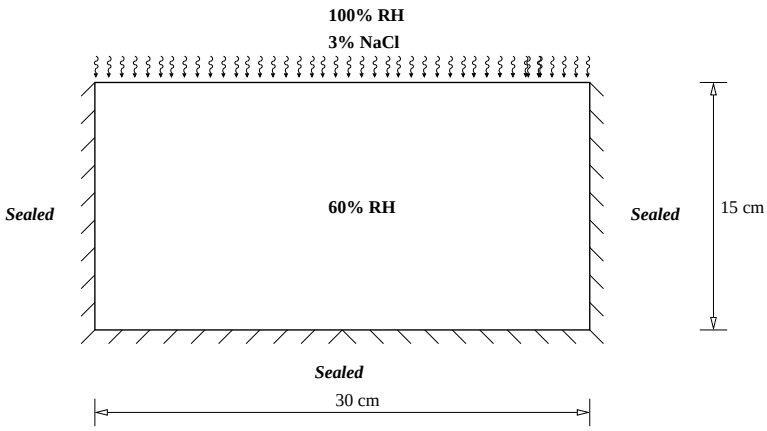

FIGURE 3. The physical model for chloride ion penetration and moisture diffusion in a concrete slab

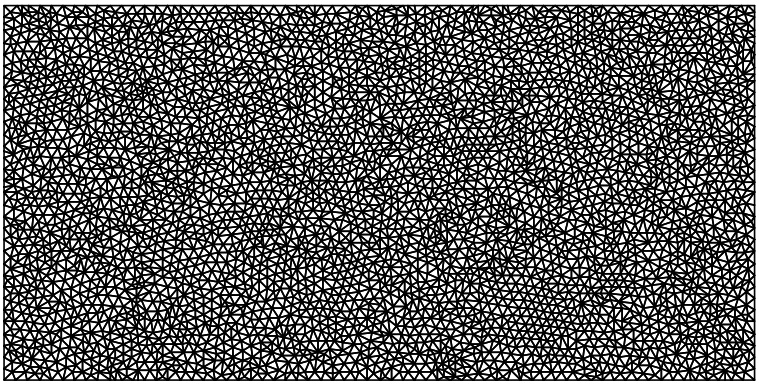

FIGURE 4. The typical finite element mesh used in the analysis for the concrete slab

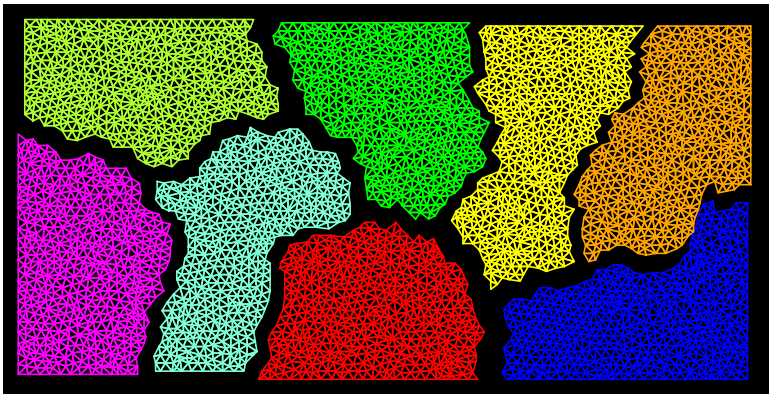

FIGURE 5. An example of mesh partitioning (partitioned into 8 sub-domains)

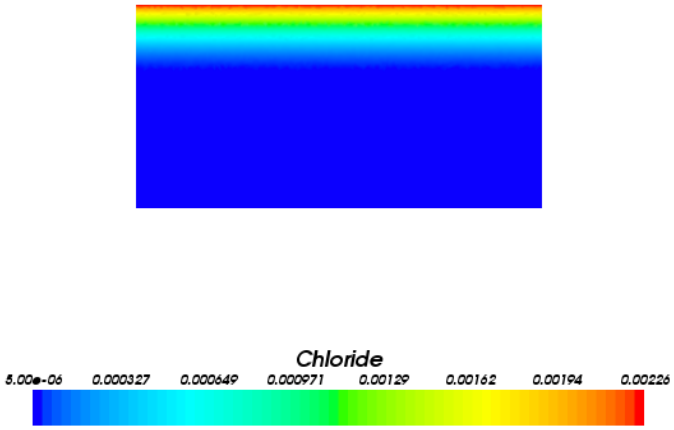

FIGURE 6. Chloride ions distribution at time = 400 days

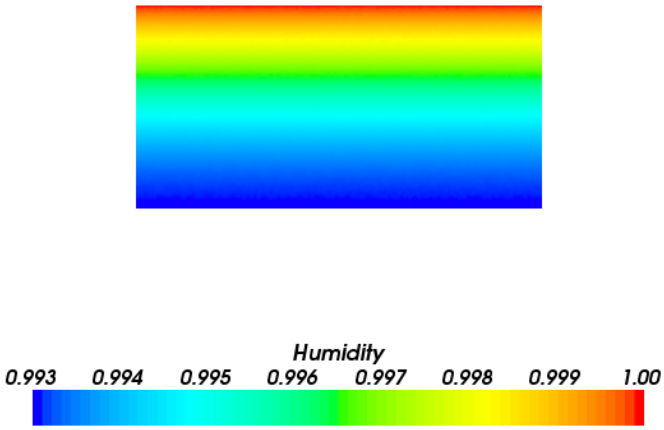

FIGURE 7. Relative humidity distribution at time = 400 days

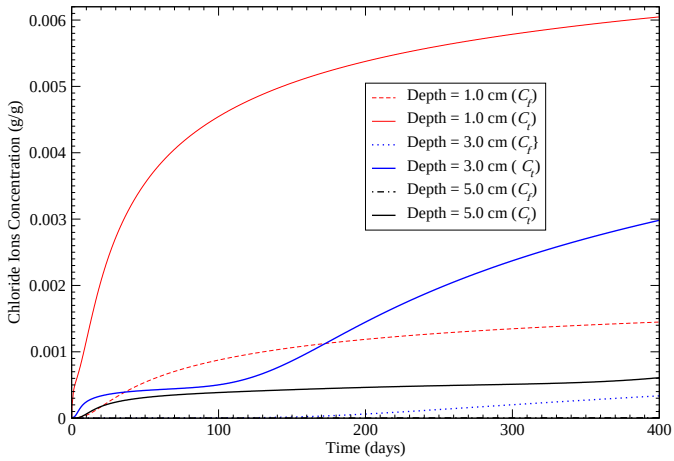

FIGURE 8. Variation of chloride ion concentration over 400 days period

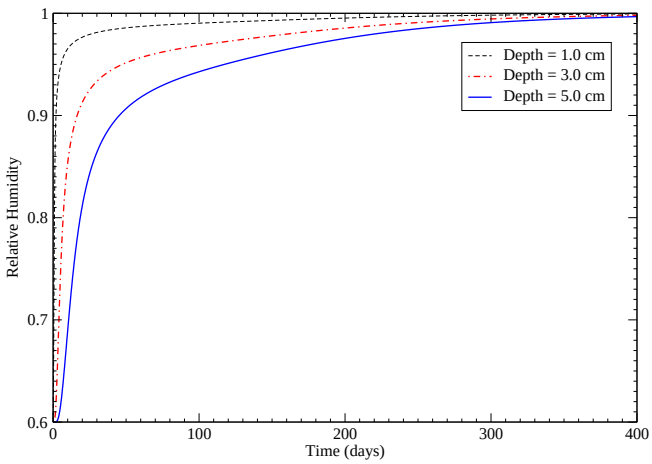

FIGURE 9. Variation of relative humidity over 400 days period

concrete slab, then it would take about 100 days for the free chloride concentration to reach the critical value on the surface of the rebar, and then the corrosion of the rebar starts. Table 4 lists a variety of suggested critical chloride contents that may initiate the corrosion of reinforcement bars in concrete. Fig. 9 shows the relative humidity profile as a function of time at different depths, i.e. at 1 cm, 3 cm and 5 cm.

TABLE 4. Suggested critical chloride ions contents

	Critical chloride ions content	**Critical chloride ions content**[**]
Berke (1986)	0.9-1.0[***]	0.039%-0.043%
Browne (1982)	0.40% (weight of cement)[*]	0.055%
FHWA	0.30% (weight of cement)[*]	0.041%
ACI (1994)	0.15% (weight of cement)[*]	0.021%
Cady and Weyers (1992)	-	0.025%-0.05%

[*] The cement content is considered as 550 lb/yd^3

[**] Total chloride content in concrete in gram of chloride per gram of concrete

[***] kg of chloride per cubic meter of concrete

TABLE 5. Performance of parallel finite element on PC clusters (Case 1: 3000 Node; Case 2: 6000 nodes; Case 3: 12000 Nodes; and case 4: 24000 Nodes)

# of Procs.	**Execution Time (sec.)**			
	Case 1	Case 2	Case 3	Case 4
1	1047	2208	5393	12698
2	1312	3833	10768	33155
4	512	1301	3990	11132
8	209	557	1515	4639
16	97	237	623	1607
32	68	123	276	709
64	55	84	135	316

Table 5, Fig. 10, Fig. 11, Fig. 12 and Fig. 13 show the performance of the parallel computation. Four meshes with the numbers of nodes of 3000, 6000, 12000 and 24000 were analyzed. In all four cases, the execution time using 2 processors is higher than that of 1 processor (see Table 5), which is actually not a surprise, because we used a direct solver, LU factorization, to solve the local problem. This approach performs better on larger number of processors in which the local problem becomes smaller. In addition, the speed of the network plays important role in the parallel computing. Up to 64 processors were used in the example.

The speed up and efficiency are shown in Fig. 10 and Fig. 11, respectively. They are based on the $T_s(n)$ taken to be the time to run the fastest serial algorithm on one processor of our parallel machine system. The speed is approaching the ideal case (linear speed up) with the increase of problem size (see Fig. 10). This is because in larger problems, the local computation time becomes longer relative to the communication time. Even though still debatable, the minimum speed up value of 20.0 is the most widely accepted as the minimum value. As for the efficiency, the

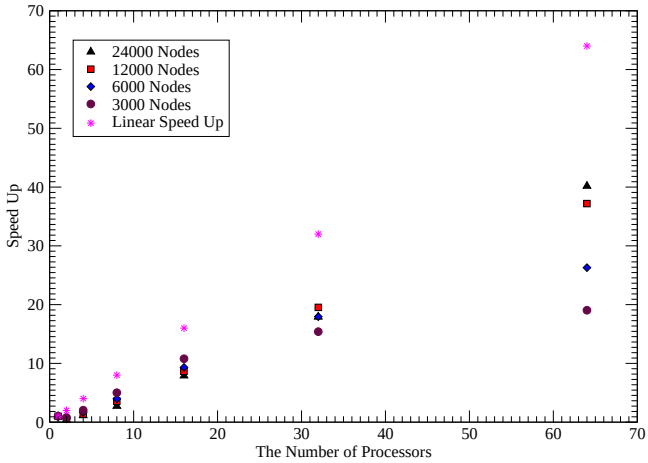

FIGURE 10. Speed up over a number of processors used in the analysis

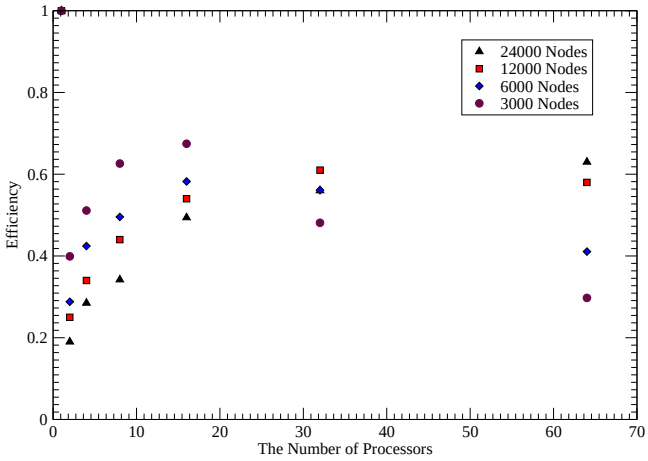

FIGURE 11. Efficiency over a number of processors used in the analysis

maximum value reached in our study is about 0.6 (the ideal would be 1.0). This is mainly due to the limited problem size.

Fig. 12 and Fig. 13 show the speed up and efficiency by replacing $T_s(n)$ with $T_2(n)$, respectively, where $T_2(n)$ is the time taken by the fastest parallel algorithm on two processors. As mention before, this comparison is necessary to see the performance of our parallel algorithm as our algorithm cannot be run on one processor. Except for the problem size of 3000 nodes, the results for both speed up

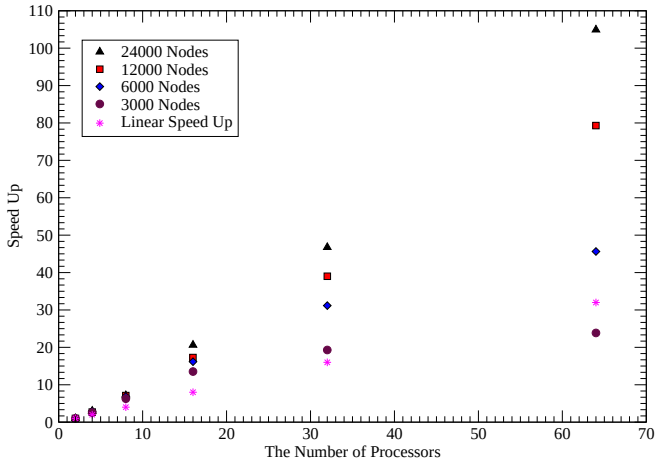

FIGURE 12. Speed up over a number of processors used in the analysis using $T_2(n)$ as reference

and efficiency are better than those ideal cases. As the problem size increases, both the speed-up and efficiency are getting better. At 64 processors, the speed-up and efficiency for the largest problem size reach 107 and 3.3, respectively. This shows that the employed parallel algorithm performs quite well for our problem.

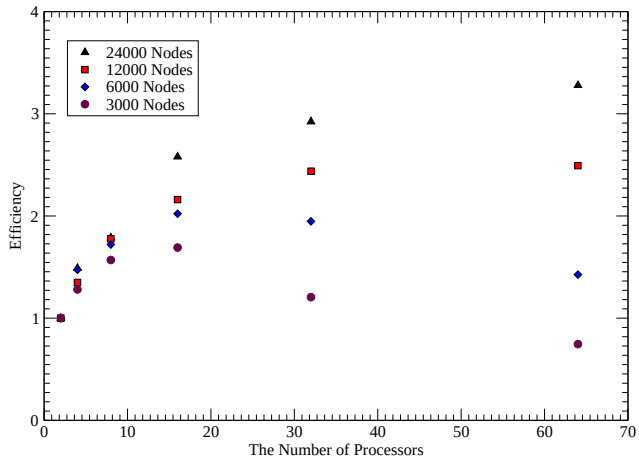

FIGURE 13. Efficiency over a number of processors used in the analysis using $T_2(n)$ as reference

As the number of processors used in the analysis increases, the size of problem solved by each processor decreases. So, the speed-up increases with increasing number of processors. This simply means that the time required for solving the problem becomes shorter when more processors are used in the analysis. But, with more processors, the communication between the processors increases, and the interface boundaries between partitions increase, which result in larger size of Schur complement system, and thus require longer computing time. These are the reasons why there will be an optimum number of processors beyond which increasing the number of processors will not improve the speed of the computation. Even though our results do not show the optimum condition, one can still see the trend that there will be an optimum condition under which the increase of number of processors will not be able to enhance the speed up. For example, the speed up of the problem with the size of 3000 nodes (Fig. 10 and Fig. 12) almost reaches the optimum condition at 64 processors.

7. Conclusion

Two fully-coupled partial differential equations are established for characterizing the chloride ion penetration and moisture diffusion in concrete. The model prediction can simulate realistically the coupled chloride penetration and moisture diffusion. Six material parameters in the two partial differential equations need to be determined in order to solve the equations numerically, including two diffusivities, two capacities, and two coupling parameters. The material models developed for the two diffusivities and two capacities are introduced. Further experimental research is needed to validate the two coupling parameters.

A parallel finite element program is developed to solve the two fully-coupled partial differential equations. The parallel algorithm to generate and to solve the finite element problem is presented in detail and the algorithm is implemented using the functions of PETSc and MPI (i.e. Portable, Extensible Toolkit for Scientific Computation and The Message Passing Interface standard). The performance of the parallel finite element program is evaluated by two indicators: the speed-up and the efficiency.

Chloride penetration and moisture diffusion in a concrete slab is used as a numerical example. The results obtained from the example show that the parallel computing program produces the same results as the single processor program, but with less time. Therefore, the parallel computing program provides a powerful tool to solve the computationally intensive problem, especially for large scale structures. The model predictions can be used for characterizing chloride and moisture distributions in concrete at any time. With given critical chloride concentration, the results can also be used to estimate the time for the onset of steel corrosion in concrete.

With increasing number of processors, the required computing time decreases, but the internal communications between the sub-domains increase, and the size of Schur complement system in the parallel algorithm also increases, which reduce the speed-up. Therefore, there exists an optimum number of processors for each specific application. Depending on the physical problem under consideration, especially on the number of finite elements used, the optimum number of processors varies. Thus, we can conclude that the non-overlapping or substructuring method is more effective on larger number of processors (before reaching the optimum number of processors).

Acknowledgments

The research was supported in part by NSF grants ACI-0112930, ACI-0305666, and CMS-4872379 to the University of Colorado at Boulder.

References

[1] A. Ababneh, *The Coupled Effect of Moisture Diffusion, Chloride Penetration and Freezing-thawing on Concrete Durability*, PhD Dissertation, University of Colorado at Boulder, 2002.

[2] A. Ababneh, F. Benboudjema, and Y. Xi, *Chloride penetration in non-saturated concrete*, Journal of Materials in Civil Engineering (ASCE), March/April: 183-191, 2003.

[3] A. Ababneh and Y. Xi, *An experimental study on the effect of chloride penetration on moisture diffusion in concrete*, Materials and Structures (RILEM), 35(254): 659-664, 2002.

[4] ACI Committee 201, *Guide to durable concrete, Chapter 4 – Corrosion of steel and other materials embedded in concrete, Manual of Concrete Practice, Part I*, 1994.

[5] C. Alonso, C. Andrade, M. Castellote, and P. Castro, *Chloride threshold values to depassivate reinforcing bars embedded in a standardized OPC mortar*, Cement and Concrete Research, 30: 1047-1055, 2000.

[6] C. Andrade and D. Whiting, *A comparison of chloride ions diffusion coefficients derived from concentration gradients and non-steady state accelerated ionic migration*, Materials and Structures (RILEM), 29: 476-484, 1996.

[7] S. Balay, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang, *PETSc Users Manual*, Report No. ANL-95/11 - Revision 2.1.5, Argonne National Laboratory, Argonne, IL, 2004.

[8] S. Balay, V. Eijkhout, W. D. Gropp, L. C. McInnes, and B. F. Smith, *Efficient management of parallelism in object oriented numerical software libraries*, In Modern Software Tools in Scientific Computing (Ed. E. Arge, A. M. Bruaset and H. P. Langtangen), Birkha user Press, 163-202, 1997.

[9] Z. P. Bazant and L. J. Najjar, *Nonlinear water diffusion of nonsaturated concrete*, Matériaux et Constructions, 5(25), 1972.

[10] N. S. Berke, *Corrosion rates of steel in concrete*, ASTM Standardization News, 14(3): 57-61, 1986.

[11] R. Browne, *Design prediction of the life for reinforced concrete in a marine and other chloride environment*, Durability of Building Materials, 1(2): 113-125, 1982.

[12] P. D. Cady and R. E. Weyers, *Predicting service life of concrete bridge decks subject to reinforcement corrosion*, In Corrosion Forms & Control for Infrastructure, San Diego, CA, November, 1992.

[13] R. M. Christensen, *Mechanics of Composite Materials*, Wiley Interscience, New York, 1979.

[14] M. Collepardi, A. Marciallis and R. Turriziani, *Penetration of chloride ions in cement paste and in concrete*, Journal of the American Ceramic Society, 55(10): 534-535, 1972.

[15] H. M. Jennings and Y. Xi, *Microstructurally based mechanisms for modeling shrinkage of cement paste at multiple levels*, In Proc. of the 5th Int. Symp. on Creep and Shrinkage of Concrete, Barcelona, Spain, September, 1993.

[16] P. K. Jimack and N. Touheed, *Developing parallel finite element software using MPI, In High Performance Computing for Computational Mechanics*, (Ed. B. H. V. Topping and L. Lammer), Saxe-Coburg Publication, 15-38, 2000.

[17] G. Karypis, *Multilevel algorithms for multi-constraint hypergraph partitioning*, University of Minnesota, Department of Computer Science / Army HPC Research Center, Technical Report #: 99-034, Minneapolis, Minnesota, 1999.

[18] N. S. Martys, *Survey of concrete diffusion properties and their measurement*, Building and fire research laboratory, NIST, Gaithersburg, Maryland, 1995.

[19] N. S. Martys, S. Torquato, and D. P. Bentz, *Universal scaling of fluid permeability for sphere packing*, Physical Review E, 50(1): 403-408, 1994.

[20] I. Masters, A. S. Usmani, J. T. Cross, and R. W. Lewis, *Finite element analysis of solidification using object-oriented and parallel techniques*, International Journal for Numerical Methods in Engineering, 40: 2891-2909, 1997.

[21] C. L. Page, N. R. Short, and A. El Tarras, *Diffusion of chloride ions in hardened cement paste*, Cement and Concrete Research, 1(3): 395-406, 1981.

[22] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, 1996.

[23] J. R. Shewchuk, *Triangle: Engineering a 2D quality mesh generator and delaunay triangulator*, In First Workshop on Applied Computational Geometry (Philadelphia, Pennsylvania), ACM, 124-133, 1996.

[24] J. R. Shewchuk, *Delaunay refinement algorithms for triangular mesh generation*, Computational Geometry: Theory and Applications, 22(1-3): 21-74, 2002.

[25] B. Smith, P. Bjørstad, and W. Gropp, *Domain Decompostion: Parallel Multilevel Methods for Ellipic Partial Differential Equations*, Cambridge University Press, Cambridge, 1996.

[26] M. Snir, S. W. Otto, S. Huss-Lederman, D. W. Walker, and J. Dongarra, *MPI the Complete Reference*, The MIT Press, Cambridge, Massachusetts, 1996.

[27] L. Tang and L. O. Nilson, *Chloride binding capacity and binding isotherms of OPC pastes and mortars*, Cement and Concrete Research, 23: 247-253, 1993.

[28] H. R. Thomas, K. Morgan, and R. W. Lewis, *A fully nonlinear analysis of heat and mass transfer problems in porous bodies*, International Journal for Numerical Methods in Engineering, 15: 1381-1393, 1980.

[29] B. H. V. Topping and A. I. Khan, *Parallel Finite Element Computations*, Sase-Coburg Publications, Edinburgh, 1996.

[30] Y. Xi, *A model for moisture capacities of composite materials – application to concrete*, Computational Material Science, 4: 78-92, 1995.

[31] Y. Xi, *A model for moisture capacities of composite materials – formulation*, Computational Material Science, 4: 65-77, 1995.

[32] Y. Xi and Z. P. Bazant, *Modeling chloride penetration in saturated concrete*, Journal of Materials in Civil Engineering (ASCE), 11(1): 58-65, 1999.

[33] Y. Xi, Z. P. Bazant, L. Molina, and H. M. Jennings, *Moisture diffusion in cementitious materials: Adsorption isotherm*, Journal of Advanced Cement-Based Materials, 1: 248-257, 1994.

[34] Y. Xi, Z. P. Bazant, L. Molina, and H. M. Jennings, *Moisture diffusion in cementitious materials: Moisture capacity and diffusivity*, Journal of Advanced Cement-Based Materials, 1: 258-266, 1994.

[35] Y. Xi and H. M. Jennings, *Shrinkage of cement paste and concrete modeled by a multiscale effective homogeneous theory*, Materials and Structures (RILEM), 30: 329-339, 1997.

Department of Civil, Environmental, and Architectural Engineering, University of Colorado at Boulder, Boulder, CO 80309, USA
E-mail: `suwitoa@colorado.edu`

Department of Computer Science, University of Colorado at Boulder, Boulder, CO 80309, USA
E-mail: `cai@cs.colorado.edu`

Department of Civil, Environmental, and Architectural Engineering, University of Colorado at Boulder, Boulder, CO 80309, USA
E-mail: `yunping.xi@colorado.edu`